
MAGAZYN

AMIGA NG

AmigaOS 4 - MorphOS - AROS

(3) 02/2018



Coś TY zrobić dla swojego NG ?

WSTĘPNIAK

Pierwszy nasz numer na papierze. Jeszcze nie wiem jak to wyszło, ile trzeba będzie zmienić, i przede wszystkim czy utrzymamy się w tej formie – czy będzie na tyłu chętnych, żeby pismo kupować.

Tematyka jest jak zwykle ciekawa i różnorodna.

Pod niepozornym tytułem „Blat Workbencha” kryje się tekst dla początkujących, opisujący praktyczne użytkowanie komputera z AmigaOS 4. Jest wszystko – od podziału dysku na partycje po omówienie programów, które warto mieć i używać. Dalej jest artykuł o uruchamianiu starego oprogramowania – również tekst dla początkujących, porządkujący temat kompatybilności i pokazujący rozwiązania typowych problemów. Następna jest mała galeria pokazująca AmigaOS 4. Potem szkołka DrawStudio – programu do grafiki wektorowej. Niestety nie weszła już szkołka AmiFIG-a. Bardzo bym chciał, żeby nasze pismo miało stałych nabywców, wtedy moglibyśmy planować i zwiększać nakład. Co za tym idzie zwiększyć objętość, bo miałem wybór: pobić objętość Amigazynu albo wyjść drukiem. To są trudne wybory. Dział OS 4 zamyka kolejną część szkoły Hollywood – tym razem odtwarzamy zegar z Workbencha. Rzecz jest pozornie błaha, bo od teraz możecie mieć amigowy zegar na każdym systemie.

Kącik MorphOS-a rozpoczyna poradnik jak samemu naprawić zasilacz w notebookach Apple. Konkretnie to opisane są dwa sposoby zreperowania urwanego kabla (dwa, bo urywa się albo od strony wtyczki, albo zasilacza) „Szaty motyla” to wskazówki jak samemu zrobić skórę dla systemu MorphOS. Jest też przegląd wybranych programów pod Mac OS X, który to system może być świetnym uzupełnieniem w dual-boocie.

AROS to z pewnością wyczekiwany samouczek jak postawić Icaros Desktop w maszynie wirtualnej. Jest też recenzja gry „Hermes” oraz opis obszernej paczki gier dla AROS-a. Jeżeli nie wiedzieliście jak zacząć z AROS-em, to ten numer magazynu jest świetną zachętą.

Na zakończenie system FriendUP, opisany z punktu widzenia dewelopera.

Tyle streszczenia zawartości. Mam nadzieję, że z każdym numerem czasopiśma będzie przybywać powodów, by te nasze komputery włączać. Liczę na waszą aktywność.

Adam Mierzwa

Spis treści

Nowości	2
Blat Workbencha	4
Uruchamianie starych programów	9
Galeria AmigaOS 4	14
DrawStudio	16
Hollywood #3	25
Jabłko pod prądem	35
Szaty motyla	38
Gdzie MorphOS nie da rady, tam Mac OS pomoże	40
Instalacja AROS-a w maszynie wirtualnej	43
Hermes	51
Aspire_games_1	53
FriendUP	58

MAGAZYN

AMIGA NG

Numer (3) 2/2018

Skład: PagePlus X9

Redakcja:

Adam Mierzwa
adammierzwa1@gmail.com

Okładka I: Konrad Czuba
Okładka IV: Rafał Chyła
(zrobiona na AmigaOne X1000)

Autorzy tekstów:

Radosław Czernik
Konrad Czuba
Aleksander Giedyk
Marcin Kornaś
Adam Mierzwa
Rafał Pączko
Paweł Stefański
Krzysztof Radzikowski

Nowości

Po niemrawej końcówce 2017 coś drgnęło w oprogramowaniu. To wszystko przenika się nawzajem, np. gra Beret została przeportowana wcześniej dla AmigaOS 4, nie wspominając jak stary jest to tytuł (w tym numerze można zobaczyć w opisie paczki gier dla AROS-a). Ale są powody do radości, jest tego sporo.

ScummVM

ScummVM to program do obsługi gier z innych platform, głównie przygodówek. Wersja 2.0 wnosi obsługę kolejnych 23 tytułów oraz wiele poprawek.

Aplikację można pobrać z oficjalnej strony projektu

scummvm.org/downloads/

bądź z OS4 Depot.

Powstała też nowa wersja dla MorphOS-a, aktualnie w wersji beta:

www.morphos-storage.net/?Page=Emulation&file=ScummVM_2.0.0_beta.lha

MUI

Nowa wersja klasy MUI NList (0.123). Skompilowana na systemy AmigaOS 3.x/4.x/MorphOS/AROS. Do pobrania na Aminiecie.

Udostępniono także MUI5-2017R4 dla systemu AmigaOS 3.x oraz AmigaOS 4.x:

muidev.de/downloads

Hollywood

Dużo nowości

- w promocji program dla jednej platformy sprzętowej (Amiga, Linux, Mac OS lub Windows) kosztuje tylko 49 euro; to dużo taniej (o połowę) niż normalna cena dla wszystkich platform (99 euro)
- wyszła nowa wersja 7.1 (dla użytkowników 7.0 aktualizacja jest darmowa)
- nowa wtyczka do odtwarzania plików dźwiękowych SID
- do wersji 7.1 zostały zaktualizowane: Player, SDK, wtyczki do Cubic IDE i UltraEdit, dokumentacja programu i SDK
- wersja demo (ograniczenia czasowe i funkcjonalne) niemniej nie było tego wiele lat

www.hollywood-mal.com/download.html

DCraw

Nowa wersja (9.27) programu do obróbki zdjęć w formacie RAW, dla MorphOS-a:

www.cybercom.net/~dcoffin/dcraw/

Git

Git to system kontroli wersji, bardzo przydatny dla programistów do prac nad wspólnym projektem. Ukazała się wersja dla systemu MorphOS:

github.com/widelec-BB/git-morphos/releases/tag/v2.16.2-morphos

ACE

Nowa wersja 1.15 emulatora Amstrada CPC/CPC+ dla MorphOS-a. Do pobrania z Aminetu lub MorphOS-Storage.

KwaKwa

Komunikator dla MorphOS-a. Opublikowano kod źródłowy programu, jak i modułów do niego:

github.com/widelec-BB/KwaKwa

github.com/widelec-BB/gg.module

github.com/widelec-BB/jabber.module

github.com/widelec-BB/gtalk.module

MilkyTracker

Nowa wersja 1.02 programu do tworzenia „modków”. Wymagana jest najnowsza wersja SDL2. Obie te rzeczy są na OS4 Depot

GrafX

Nowa wersja programu graficznego dla AmigaOS 4, wzorującego się na takich pozycjach jak „Deluxe Paint” i „Brilliance”. Port zrobiono w oparciu o najnowsze źródła wersji 2.5 WIP. Do pobrania na OS4 Depot.

Wysyp portów

Prawdziwy urodzaj portów SDL, większość to gry na MorphOS-a: **Beret 1.2.1**, **OpenJazz**, **XRick 2.0.0**, **CadogGL**, **Secret Mayro Chronicles**, **SDL-Ball 1.0.2**, **MiniSlug**, **Ltris**, **Hocoslamfy**, **ClassicInvaders**, **C-Dogs_0.5.8**, **Stunt Car Racer Remake** i **TCGS Cave** (wcześniej były wersje na AmigaOS 4), **Stratagus** (silnik gier RTS) w wersji 2.4.2. Z rozbudowanych pozycji: **BosWars 2.7.0** (RTS), **Naev 0.7.0** (rozgrywka w kosmosie) i **Wyrmsun 3.3.0** (RTS czerpiący z „Warcraft II”). Do pobrania z MorphOS Storage.

BOH

Ukazało się 18. uaktualnianie tej gry, obejmujące m.in. wersje dla AmigaOS 4 i AROS-a. Od teraz gra nazywa się „BOH Advance”.

AmiSSL

Nowa wersja biblioteki (4.2). Kompatybilna z OpenSSL 1.1.x, rzecz niezbędna do współczesnego korzystania z internetu. Do pobrania z Aminetu.

Spencer

Ukończono prace nad grą, zaplanowany termin wydania to marzec. W tej chwili nie jest znana cena, ale tytuł jest już oficjalnie na Steamie. Spencer to prosta platformówka 3D, ale wersja dla AmigaOS 4 jest przełomem, z powodu wykorzystania możliwości współczesnych kart graficznych.

entwickler-x.de/spencer

REBOL

Po długiej stagnacji Sassenrath zabrał głos – będzie aktualizacja strony projektu, uwolnienie Rebol/Command i powiązanych komercyjnych produktów. Przyszłość jest związana z „chmurą” i odgałęzieniem REBOL-a o nazwie Red.

www.red-lang.org/2018/03/red-rebol-carl.html

AROS na SAM

Po latach powrócono do kompilacji AROS-a na komputerki SAM: aros.sourceforge.net/nightly1.php

X500 Pro

Loriano Pagni i jego kolejne podejście do obudowy „wszystko w jednym” (poprzednie to X500 Plus). Tym razem zamiast aluminium jest plastik, wprowadzono wiele usprawnień.

Obudowa może pomieścić płytę główną mini ITX (17x17cm), zasilacz Flex-ATX i kartę graficzną dwuslotową do 170 mm długości. X500 Pro zawiera podświetlaną klawiaturę mechaniczną Ducky z serii Zero; opcjonalnie można dokupić dwa wiatraki 50 mm, mocowanie karty 16x PCI-Express i wiele możliwości personalizacji. To jest ciekawy nurt konstrukcji, będący hołdem dla komputerów domowych lat 80. i 90.

Produkt był wyczekiwany, ale z niejasnych powodów zbiórka na Kickstarterze została zamknięta po tygodniu i anulowana. Niemniej warto zapoznać się z zawartością strony projektu:

www.kickstarter.com/projects/505470364/the-x500-pro-computer-case



Blat Workbench

radzik

Mamy czas kryzysu w świecie Amigi NG. Wydaje się, że rozwój Nowej Generacji wyhamował. Panuje powszechny brak wiary w sens dalszej pracy i ogólne zniechęcenie – zarówno po stronie użytkowników, jak i deweloperów. Jednocześnie mamy czas oczekiwania na AmigaOne 1222 i nadziei, że ten tani, masowy model, technie nowe życie w społeczność, i dostarczy nowych sympatyków AmigaOS 4 i dalszego postępu dla Amigi.

Spośród wielu pomysłów, które mają na celu propagowanie AmigaOS 4, prym wiodą różnego rodzaju recenzje programów i gier. Są też artykuły publicystyczne, historyczne, a niektórzy wręcz przekonują na siłę. Według mnie brakuje rzeczy oczywistej, a zarazem najprostszej: pokazania w jednym artykule jak korzystać z AmigaOne i AmigaOS 4. Co na tym sprzęcie można zrobić?

Pierwszą rzeczą, jaka przychodzi do głowy, jest zaprezentowanie, jak wygląda to na mojej konfiguracji (AmigaOne X5000 i AmigaOS 4.1 FE). Jak został podzielony dysk, jak wygląda blat Workbench, jakich programów używam najczęściej. Planowana jest też część druga, w której opiszę jak zorganizowałem swój warsztat pracy. Taki „Radzik od kuchni”, gdzie pokażę swój zestaw narzędzi

związany z pisanem artykułów i książek, przeprowadzaniem wywiadów tekstowych, nagrywaniem podcastów oraz robieniem filmów na YouTube.

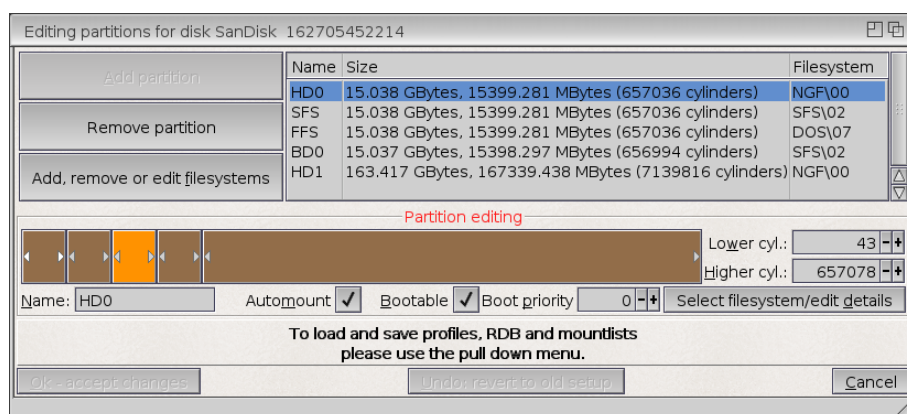
Tymczasem, zaczynamy od **podziału na partycje**:

- Workbench41FE,
- Workbench41FE_bck,
- SFS_Data,
- FFS_Data,
- Work,
- Data.

Istotna uwaga: nie jest możliwa konwersja typów partycji bez utraty danych.

Tak więc już na etapie instalacji systemu, podział dysku musi być rozsądny i przemyślany. Nie musimy od razu tworzyć wszystkich partycji, ale takie rzeczy jak typ plików partycji systemowej, czy ten na backup danych jest istotny i praktycznie niezmienny.

Workbench41FE to po prostu moja partycja systemowa. Na niej mam zainstalowany AmigaOS 4.1 Final Edition. Z racji, że lubię testować nowe rozwiązania, a dodatkowo dla komputera AmigaOne X5000 polecany jest system plików NGFileSystem, zdecydowałem się, iż pierwsza partycja (HD0), będzie właśnie z niego korzystać. Alternatywą byłoby użycie znanego od lat FastFileSystem 2, który jednak jest mało wydajnym rozwiązaniem. Natomiast SmartFileSystem 2, oficjalnie, w najnowszej wersji systemu nie jest obsługiwany. Aby korzystać z jego dobrodziejstw, trzeba posiadać Enhancer Software (pakiet ten jest dostarczony do każdego komputera AmigaOne X5000). Instalacja jednak jest dość skomplikowana, a więc (szczególnie dla początkujących użytkowników AmigaOS) nie zachęca do użycia SFS2 na głównej partycji.



Workbench4IFE_bck (jest to dysk ukryty poprzez preferencje Workbench'a) to partycja stworzona specjalnie dla programu BackUp. Dzięki temu kopia systemu tworzona jest na dysk (BD0), który ma włączoną opcję Bootable. Oznacza to, że w razie co, jesteśmy w stanie uruchomić system z kopii zapasowej. Aby mieć pewność zachowania danych, został użyty system plików SFS2.

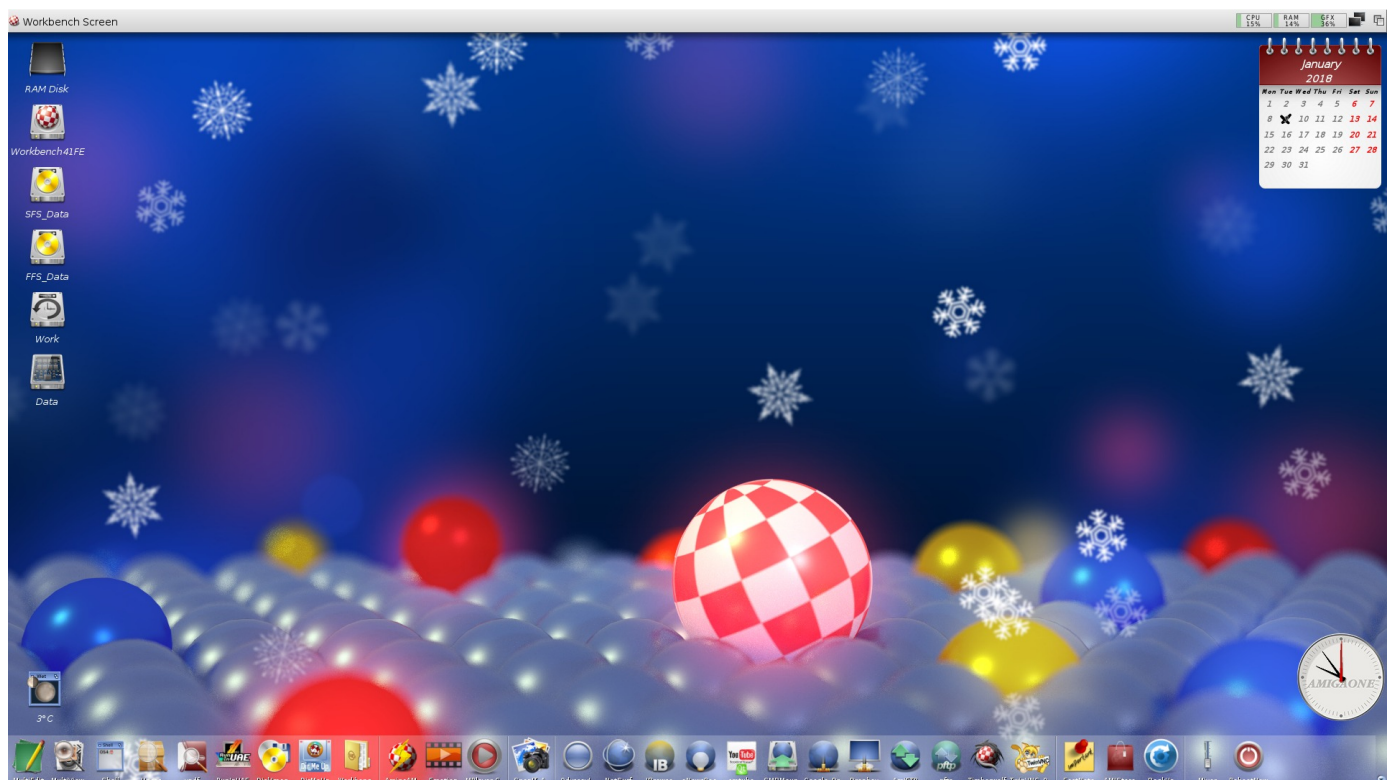
SFS Data i *FFS Data* to partycje utworzone dla programów, które lepiej działają na starszych systemach plików. Jest kilka takich opornych pozycji, między innymi: Timberwolf, eNewsReader, czy Odyssey Web Browser. Niestety NGFileSystem jest cały czas w wersji beta. Może się więc trafić oprogramowanie, które „gubi” plik pod NGFS. Gdy coś takiego nastąpi, warto mieć dodatkową partycję z bezproblemowym i sprawdzonym SFS2 lub FFS2.

Partycje *Work* i *Data* (można też stworzyć po prostu jedną dużą) służą odpowiednio do pracy, jak i gier; oraz jako magazyn dla archiwów, plików multimedialnych itp. W przypadku *Work* system plików to NGFS, a dla partycji *Data* to SFS2.

Wspomniałem już, iż FastFileSystem 2 jest bardzo mało wydajnym rozwiązaniem. O ile w czasach, gdy dyski były małe, a dane „lekkie”, FFS zdawał egzamin doskonale. W obecnych czasach, gdzie wszystko liczone jest w GB, nie spełnia on pokładanych w nim nadziei. Jednym sensownym rozwiązaniem dla tego systemu plików jest użycie go do kopii bezpieczeństwa lub kompatybilności z niektórym oprogramowaniem. NGFileSystem jest obecnie najnowocześniejszym, aktywnie rozwijanym system plików dla AmigaOS. Również względy wydajnościowe przemawiają za NGFS lub SmartFileSystem 2 (gdy

potrzebna jest zgodność i szybkość jednocześnie).

Wspomnijmy o wielkościach poszczególnych partycji. W czasach AmigaOS 3.x tworzyło się partycję dla systemu, która była stosunkowo mała, oraz dodatkowo jedną (lub kilka) dla programów, gier oraz danych. W świecie AmigaOS 4.1 wygodniej jest utworzyć po prostu jedną dużą partycję (np. 15 GB) dla systemu, dodatkową o podobnej wielkości w formacie SFS2, dla starszych programów oraz trzecią (o tej samej wielkości co partycja systemowa) na kopię zapasową. Jest to oczywiście przykładowy podział, choć w przypadku NGFS zalecane jest posiadanie partycji SFS/FFS celem uniknięcia wspomnianych problemów z np. znakomitą Odyssey. Spytać się też, czemu aż 15 GB? Dlatego, iż programy dla najnowszego AmigaOS urosły, a dodatki takie jak Qt, czy AmiCynix zajmują sporo miejsca. Ponadto, dzięki temu bez problemu



będzie można instalować programy na partycji dla systemu, i przy tworzeniu kopii zapasowej od razu mieć kompletny backup środowiska pracy.

Jeśli chodzi o **blat Workbench**, to jest on u mnie czysty. Oprócz ikon napędów mam tylko w narożnikach kalendarz i zegar z pakietu Enhancer Software. Można je dowolnie dokować na pulpicie, ja zorganizowałem to tak, żeby ogarniać je wzrokiem, ale jednocześnie, aby te dodatki mnie nie rozpraszały. Przydatnym narzędziem jest również Wet. Program, który na bieżąco informuje o pogodzie. W moim przypadku wybrałem lewy dolny róg jako pozycję dla ikony z informacją o temperaturze. Wet może też być umieszczony w Dock oraz skorzystać z kilku dostępnych skórek.

Górną belkę Workbench wzbogaciłem o graficzne wskaźniki z pakietu zTools (do nabycia w AMISore, o czym później), pokazujące mi podstawowe parametry „życiowe” systemu. Mówimy tu o obciążeniu procesora, zajętej pamięci RAM czy wykorzystanie pamięci karty graficznej. Pchełki nazywają się CPUdock, RAMdock, GFXdock. Dodatkowo używam Netdock (również z pakietu zTools), który pokazuje aktywność połączenia sieciowego. Umieszczenie wspomnianych wskaźników, to w zasadzie trik wizualny. Wskaźniki jako takie nie znajdują się w belce, lecz nad nią. Realizację tego typu „oszustwa” umożliwia Amidock. Skrótowo rzecz ujmując, wystarczy stworzyć dodatkowy dock (w preferencjach Amidocka), w którym mieszczą się wybrane monitory systemu. Następnie pozycjonuje się taki

dock w wybranym miejscu na ekranie. W opcjach wizualnych wyłącza się belkę przeciągania, włącza się przezroczystość, czego efektem jest wrażenie jakby informacje znajdowały się na belce Workbench.

AmiDock to bardzo potężne narzędzie. Filozofia działania pozostaje bardzo podobna do tej z AmigaOS 3.9, jednak elastyczność i wygoda oraz opcje konfiguracji otwierają przed użytkownikiem AmigaOS 4.1 zupełnie nowe możliwości.

Poniżej przedstawię jakie to podstawowe programy zostały umieszczone w moim głównym docku, który znajduje się na dole ekranu:

- **MultiEdit** – edytor tekstu z pakietu Enhancer Software. Może on zastąpić systemowy NotePad. Funkcjonalnie, jest to bardziej rozbudowane narzędzie, ma różne dodatki w stylu pracy na zakładkach, a co więcej integruje się z rozszerzeniem ClipViewer. Ten ostatni pochodzi również od A-EON i jest menadżerem schowka,
- **MultiViewer** – następca znanego wszystkim MultiView. Podobnie jak narzędzie opisane wyżej, oferuje większe możliwości niż jego pierwowzór. Dzięki MultiViewer zapoznamy się z plikami tekstowymi, czy graficznymi. Praca również odbywa się w zakładkach, dodatkowo można dany plik otworzyć w zdefiniowanym edytorze. Oczywiście i ta aplikacja współpracuje z ClipViewer, jak i systemem Datatypes,
- **Shell** – dostęp do AmigaDOS. Dla najnowszego AmigaOS konsola uległa usprawnieniom, można zdefiniować choćby kolorystykę oraz wprowadzano pracę na modnych zakładkach,

- **Unarc** – bardzo wygodne i proste, systemowe narzędzie, służące do rozpakowywania archiwów LHA, ZIP itd. Unarc na moim systemie (co jest standardowym ustawieniem) jest używany jako Default Tool dla archiwów. Mimo wszystko dość wygodnie jest mieć go pod rękę, o czym poniżej,

- **vpdf** – przeglądarka plików PDF. Jest to nowy program dla AmigaOS (ale istniejący od lat na systemie MorphOS). Jego przewagą nad systemowym AmiPDF jest szybkość. Wykorzystując elastyczność OS 4, AmiPDF (notabene najnowsza wersja jest również dostarczona z Enhancer Software) używam jako standardowy program do obsługi plików PDF (tzw. Default Tool), a na pasku Amidock mam opisywanego vpdf. Dlaczego? Amidock pracuje w ten sposób, że dany plik (tu plik PDF) można przeciągnąć na ikonę celem wywołania akcji (np. uruchomienie vpdf i pokazanie przeciągniętego pliku),

- **RunInUAE** – dodatek (z sekcji Extras płyty instalacyjnej systemu), który ułatwia obsługę głównie gier (choć i niekiedy programów), które nie są napisane zgodnie z systemem. Oznacza to, iż właściwie większość gier dla Amigi 500/1200 uruchomicie w wygodny sposób: wykonując dwuklik na pliku ADF/DMS lub ikonie gry, zainstalowanej przy pomocy pakietu WHDLoad,

- **DiskImageGUI** – dostarczony, również poprzez „Extras”, umożliwia obsługę obrazów płyt CD, czy DVD (np. pliki ISO), ale też obrazów dyskiety (np. ADF),

- **DirMeUp** – ciekawy menadżer plików, tworzony poprzez Boing Attitude, który na rynku amigowym obecny jest od lat. Program jest płatny, choć nie oferuje nad-

zwyczajnych możliwości. Filozofia jego działania przypomina rozwiązania znane eksploratora MS Windows,

- **WorkbenchExplorer** – to konkurent dla wyżej wymienionej aplikacji. Jest zdecydowanie bardziej rozbudowany, oferuje pracę na zakładkach, i jak na świeżą aplikację robi bardzo dobre wrażenie. Niebanalną zaletą jest to, że WorkbenchExplorer jest całkowicie bezpłatny. Jeśli jednak lubicie starego dwupanelowego Directory Opusa, powyższe dwie propozycje menadżerów plików mogą być dla was tylko ciekawostką,

- **AmigaAMP** – wielu użytkowników preferuje TuneNet (obecnie część Enhancer Software), dla mnie jednak bardziej ascetyczny wygląd AmigaAMP-a jest jego największą zaletą. Zwykle i proste narzędzie, służące po prostu do słuchania MP3, ale także WAV, a nawet plików MOD, czy XM. Dzięki obsłudze pluginów TuneNeta, w zakresie odtwarzanych formatów, możliwości są praktycznie nie ograniczone,

- **Emotion** – program, który miał swoją premierę w ubiegłym roku. Jest to schludny i bardzo wygodny odtwarzacz plików video. Aplikacja jest płatna (można ją nabyć w AMIStore), wydatek rekompensuje prostotą obsługi. Zauważyć warto, że Emotion z racji mniejszej wydajności od darmowego Mplayera skierowany jest do użytkowników maszyn typu (co najmniej) AmigaOne X1000,

- **Mplayer-GUI** – bardzo przydatny dodatek do (praktycznie) dowolnej wersji MPlayera (obecnie najnowszą i polecaną jest MikkJT-MPlayer). Dzięki temu, uzyskujemy dostęp do bogatego (choć nie aż tak przemyślanego, jak w Emotion) interfejsu użyt-

kownika, który usprawnia odtwarzanie filmów, zarówno w postaci plików, jak i płyt DVD,

- **SnapIT** – program służący do robienia zrzutów ekranu. Tak jak vpdf, wywodzi się z systemu MorphOS, oferując całkiem duże możliwości, choć nie wyrastające ponad starego i dobrego (oraz stabilniejszego) SGraba.

- **Odysey** (oraz OdyseyLauncher2) – najlepsza przeglądarka WWW. Mimo sporego zastoju w rozwoju, nadal jest to program deklasujący konkurencję (w świecie AmigaOS). Poza takimi standardami, jak obsługa HTML5, zakładek, filmów na YouTube, Odysey można skonfigurować jako np. Firefox dla wybranych stron, tak by zyskać maksymalną kompatybilność,

- **NetSurf** – lekka i całkiem zgrabna, choć cały czas nie do końca dopracowana przeglądarka HTML4. Program jest aktywnie rozwijany i można go na ten czas pozycjonować jako coś pomiędzy Odysey a IBrowse. Oznacza to, że aplikacja jest dużo bardziej zaawansowana od starego IBrowse, ale jednocześnie nie jest w stanie w pełni zastąpić Odysey,

- **IBrowse** – to kultowa przeglądarka WWW dla AmigaOS. W przypadku systemów 4.1 służy ona właściwie tylko do obsługi OS4Depot i Aminet,

- **eNewsReader** – bardzo dobry czytnik RSS. Aplikacja jest darmowa, napisana została w ubiegłym roku (a więc jest to świeży program) i obsługuje import RSS ze znanej usługi Feedly. Po odpowiednim skonfigurowaniu może służyć jako świetny agregator treści, jaki i wygodny program do obsługi subskrypcji na YouTube,

- **smTube** – jeden z czołowych i lepszych programów dla fanów

YouTube. Potrafi bez problemu odtworzyć filmy z tego serwisu, możemy je również pobrać na dysk. Aplikacja pracuje w środowisku Qt, a jej rozwój jest dość systematyczny,

- **SMBMounter** – dodatek do Samby. Ułatwia konfigurację (poprzez GUI) zasobów w sieci lokalnej. Szczególnie wygodne narzędzie w przypadku posiadania dysków sieciowych,

- **Google_Drive** – skrypt napisany przeze mnie do obsługi usługi Google Drive, czyli chmury plików. Jest to możliwe dzięki Amiga Cloud Handlers. Skrypt służy do podpięcia dysku Google poprzez wybranie ikony na AmiDock,

- **Dropbox** – dokładnie to samo co wyżej, tyle że dla usługi Dropbox,

- **AmiFTP** – prosty, a przez to nie zawsze wygodny klient FTP, otwierający okno na ekranie Workbench. Służy do prosty i szybkich operacji typu „drag'n'drop”,

- **pFTP** – rozbudowana aplikacja do obsługi protokołu FTP. Program otwiera swój ekran, a jego interfejs oparty jest na schemacie dwupanelowym. Przez to, obsługa wielu plików oraz bardziej zaawansowane operacje są dużo łatwiejsze do realizacji,

- **Timberwolf** – legendarna przeglądarka Firefox (w wersji 4) dla AmigaOS 4.1. Program potrafił również skorzystać z dodatków dla przeglądarki. Niestety od lat nie rozwijany. Aplikacja jest niekiedy przydatna np. przy wgrywaniu filmów na serwis YouTube,

- **TwinVNC** – prosty i funkcjonalny klient VNC. Dzięki niemu prosto z AmigaOS można się podłączyć do komputera pracującego pod kontrolą najbardziej popularnych systemów operacyjnych. W

moim przypadku używany głównie do obsługi Raspbbery Pi,

- **FastNotes** – żółte (choć niekoniecznie), karteczki. Przydatne do robienia notatek, kopiowania fragmentów tekstu itp.

- **AMISore** – dostęp do amigowego „App Store”. Aplikacja napisana w Hollywood działa całkiem zgrabnie i umożliwia zakupy gier, programów dla AmigaOS. Między innymi, w tym sklepie można nabyć elektroniczną wersję Enhancer Software. W przyszłości AMISore powinien być rozszerzony o multimedia oraz książki,

- **BackUp** – proste, aczkolwiek skuteczne narzędzie do tworzenia kopii zapasowej partycji systemowej. Program potrafi kopiować „soft links”, tylko nowsze pliki, czy tworzyć logi.

- **Mixer** – (dodatek Extras z płyty instalacyjnej AmigaOS) panel do obsługi karty muzycznej. Umożliwia sterowanie głośnością, mikrofonem, czy źródłami dźwięku podłączonymi do komputera,

- **RebootNow** – narzędzie służące do wykonania miękkiego lub twardego resetu, bez użycia klawiatury, czy przycisku na obudowie komputera.

Oczywiście aplikacje, które znajdują się w moim AmiDocku, nie są jedynymi, jakich używam. Jest ich więcej. Niektóre z nich są podpięte pod menu kontekstowe (np. **FastView** – przeglądarka obrazków), czy wywoływane dwuklikiem na blacie Workbench (**Filer** – menadżer plików).

Elastyczność AmigaOS, zawsze pozwalała na dość swobodną konfigurację systemu. Najnowszy Workbench, w porównaniu do poprzedników (z systemów 3.x) jest dużo bardziej rozbudowanym narzędziem, które mimo wszystko, zachowuje swój pierwotny charakter. Gdybyście woleli jednak coś bardziej zaawansowanego, nic nie stoi na przeszkodzie, aby swój system wyposażyc w kultowego Directory Opusa 5.

AmigaOS używam od lat dziewięćdziesiątych. Natomiast jego najnowszą wersję polubiłem, gdy na swojej Amidzie 1200, wyposażonej w PowerPC, z ciekawości zainstalowałem wersję 4.0. Od tego momentu, gdy myślę o pracy z wykorzystaniem amigowego oprogramowania, wybieram AmigaOS 4.1. Oczywiście, mam i lubię klasyczne rozwiązania w postaci AmigaOS 3.9. Po tylu latach jednak każdemu z was polecam przejście na najnowszy dostępny system. To po prostu ewolucja. Mimo że w dzisiejszych czasach, swoimi Amigami nie zastąpimy blaszaków, jestem przykładem tego, że na AmigaOS 4.1 da się całkiem sporo zrobić.

REKLAMA



Uruchamianie starych programów

truman burbank

Problem kompatybilności ze starszym oprogramowaniem jest coraz rzadziej podnoszony. O ile kiedyś to był problem – z czasem jego dolegliwość spadła. Mimo powolnego rozwoju, mimo wszystko, jakieś nowsze oprogramowanie powstaje. A przecież jeśli ktoś decyduje się na rozwiązanie NG – oczekuje nowszego oprogramowania.

Jest też możliwość emulacji, która jest coraz lepsza (wydajniejsze konfiguracje, aktualizacje emulatorów).

Niemniej sam osobiście miałem niedosyt, jeśli chodzi o znajomość zagadnienia kompatybilności. Albo natrafiałem na obiegowe uogólnienia typu „pod OS 4 jest UAE” albo było „ślizganie się po temacie”. Jest trochę wnikliwych publikacji, np. cykl artykułów „Klasyczne gry na Amidze One X1000” na blogu Rafała Chyły, czy też niedawny artykuł Krzysztofa Radzikowskiego (w „Amigazynie” nr 14) o RunInUAE. Brakuje mi takiego całościowego ujęcia tematu w inny sposób.

Przedstawię emulację zaszytą w systemie, pokażę jak używać promotorów ekranu, będzie o emulacji wybranych chipsetów, wspomnę też o emulacji z użyciem UAE. Wszystko to oparte na praktycznych przykładach użycia.

Wprowadzenie

Jednym z założeń powstania AmigaOS 4 było niezależenie się od chipsetów. Coś, co kiedyś było dumą Amigi u jej początków, z czasem stało się balastem. Stało się oczywiste, że nie da się w nieskończoność bazować na przestarzałych układach, a zaprojektowanie i wdrożenie nowych było nierealne. Tendencje światowe były jasne – zresztą nawet teraz patrząc wstecz na bieg historii, widzimy, że unikalne rozwiązania (np. takich firm jak Silicon Graphics) nie przetrwały.

Nie da się ukryć, że mimo przestarzałości chipsetów dorobek Amigi (baza oprogramowania) jest olbrzymia. Nie można było tak po prostu zignorować to i zrobić komputer z systemem całkowicie niekompatybilny wstecz. To byłaby pełna egzotyka, produkt dla nikogo, bo raczej nie zanosilo się na to, żeby „wielcy” wspierali Amigę tak jak dawniej.

Emulacja zaszyta w systemie

Niestety nic ciekawego OS 4 u swego zarania nie miał do zaoferowania w materii kompatybilności. Jest w systemie „BlackBox”, który jest emulatorem prostym i dokładnym. Ale prawdziwym przełomem było wprowadzenie do systemu emulacji o nazwie **Petunia**.

Project Petunia



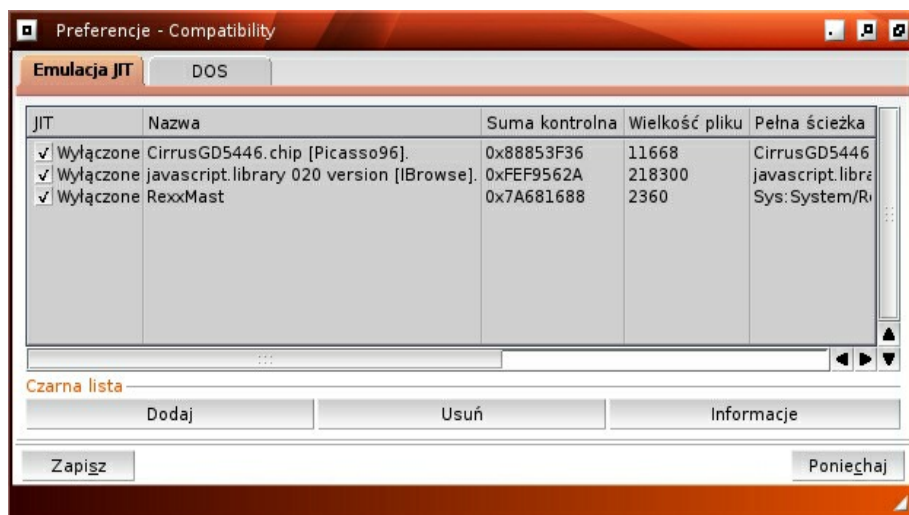
Autorem jej jest Álmos Rajnai, który przy pomocy wielu osób stworzył emulator procesora Motorola 68040 bez MMU (w systemie przedstawia się jako 68020 z koprocesorem matematycznym 68881). Jest to emulator JIT (just-in-time), inne określenie to „dynamiczna rekompilacja”, czy działanie „w locie”. Jest to inna zasada niż w BlackBoxie, który oparty jest na rekompilacji interpretacyjnej. O całym projekcie można poczytać tutaj:

<http://amigos.amiga.hu/rachy/petunia.html>

Tam wyjaśnione są dokładnie te różnice oraz podane wiele szczegółów technicznych.

Petunia jest oficjalnie częścią systemu od 2006 roku. Jak z niej korzystać?

Petunia to plik *petunia.library.kmod* znajdujący się w katalogu Kickstart, a więc ładuje się wraz ze startem systemu. Domyślnie działa zawsze, z wyjątkiem aplikacji dołączonych do „czarnej listy”.



Zrzut 2

Lista to plik tekstowy, ale dla wygody zrobiony jest panel, gdzie łatwiej można dodawać i usuwać pozycje. Już przy pierwszym zajrzeniu tam zauważymy, że pola nie są puste; wstępnie jest to skonfigurowane, są tam wpisane określone programy (choć nie ma ich wiele). Narzędzie nazywa się *Compatibility* i jest w *Prefs* (Zrzut 2). Pierwsza zakładka *Emulacja JIT* określa, które programy mają nie korzystać z emulacji. Zakładka *DOS* pokazuje sposób obsługi niekompatybilnych programów.

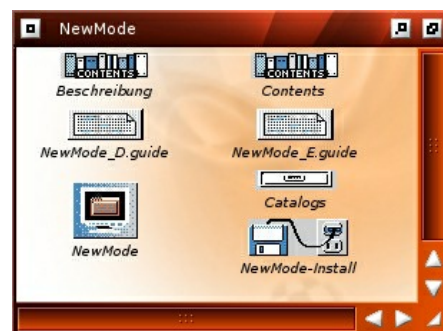
Niestety Petunia nie załatwia wszystkiego. Jeżeli program jest napisany w taki sposób, że nie odwołuje się do układów specjalizowanych – jest dobrze. Na przykład programy graficzne zazwyczaj nie potrzebują dźwięku, a do obsługi wyświetlania używają karty graficznej – w ten sposób bez żadnych zabiegów pod AmigaOS 4 bezproblemowo działa TV Paint.

Niestety nie działa to zawsze – zdarza się sytuacja, że jakiś program otwiera się w rozdzielczości, która niekoniecznie nam pasuje. Takim programem jest DrawStudio, który domyślnie chce

otworzyć się w rozdzielczości PAL: 640:400 High Res Laced. W takiej sytuacji trzeba program „wypromować”, czyli zmusić do otwarcia się w większej rozdzielczości.

Promotory ekranu

Ja ze swej strony polecam **New Mode**. Pobrane archiwum po rozpakowaniu wygląda jak na obrazku (Zrzut 3).

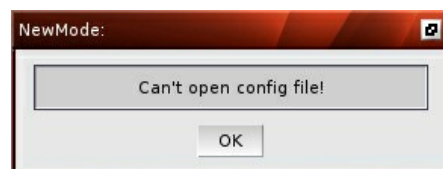


Zrzut 3

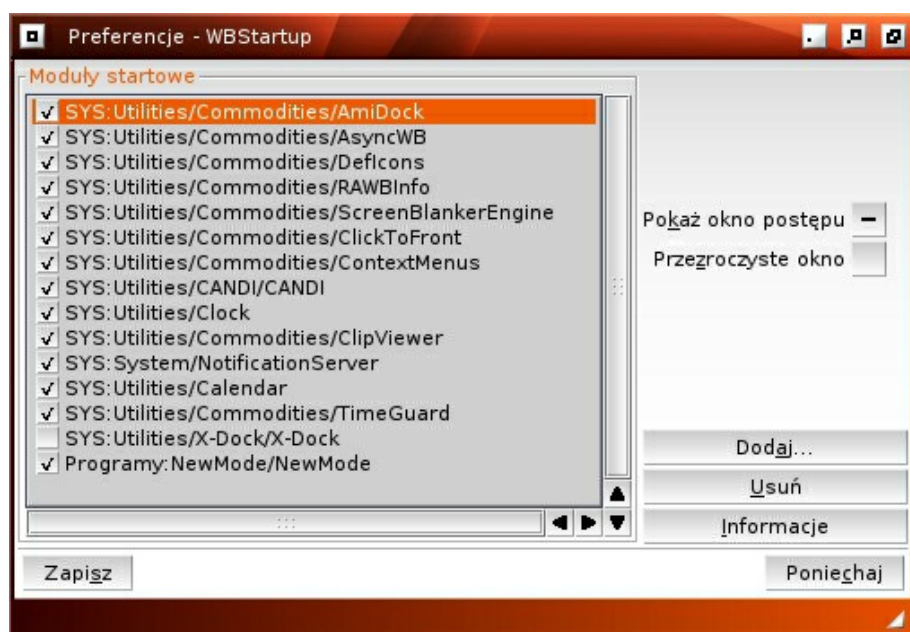
Potrzebujemy z niego tylko plik **NewMode**, który najlepiej żeby łąadował się przy starcie systemu. Najwygodniej zrobić to w ten sposób, że przeciągamy plik na otwarte okno *WBStartup* (narzędzie jest w *Prefs*). **NewMode** automatycznie łąduje na koniec listy (Zrzut 4).

Jeszcze klik w *Zapisz* i miękki reset.

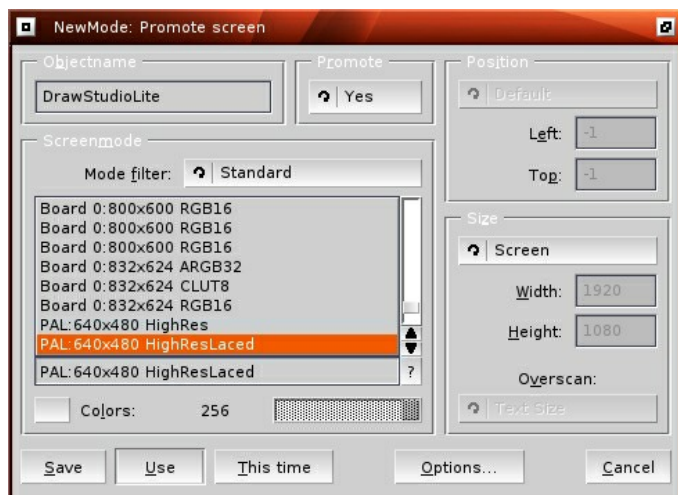
Po resecie będzie komunikat jak niżej (Zrzut 5). To normalne po pierwszym uruchomieniu.



Zrzut 5



Zrzut 4



Zrzut 6

Teraz uruchommy kłopotliwy program.

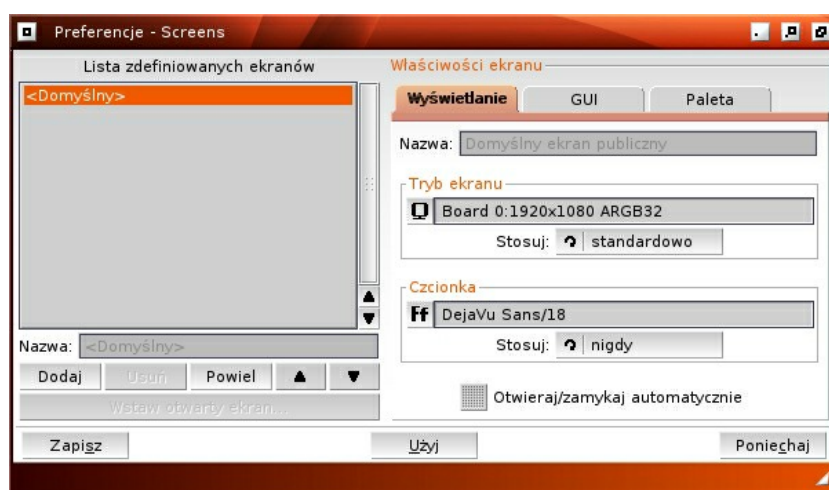
Po uruchomieniu DrawStudio wyskoczy zgłoszenie programu NewMode (Zrzut 6).

Z listy wybieramy natywną rozdzielczość monitora. Zaznaczamy też pole *Colors* i przeciągamy suwak dla zwiększenia liczby kolorów. Zapisujemy poprzez *Save* i gotowe.

Gdybyśmy chcieli zmienić parametry, NewMode można zawsze wywołać kombinacją klawiszy Alt + n (Zrzut 7).



Zrzut 7



Zrzut 8

Gwoli uczciwości trzeba wspomnieć, że jest rozwiązanie systemowe o nazwie **Screens** (Zrzut 8). Narzędzie znajduje się w Prefs. Moim zdaniem, NewMode – mimo że napisany w 1995 r. – jest rozwiązaniem lepszym, ale są sytuacje kiedy można użyć Screens. Ja tak zrobiłem przy okazji ArtEffect 4. To program graficzny, który może pracować w trybie własnego ekranu (na „ściągającym ekranie”) oraz na ekranie Workbencha (czyli menu będzie porozkładane na blacie w postaci „pływających pasków”).

Uruchamiamy ArtEffect, a po nim preferencje Screens. Powinna tam uaktywnić się pozycja „Wstaw

otwarty ekran” – wybieramy z listy „ArtEffect”. W zakładkach „GUI” i „Paleta” można przestawić „Ustawienia” na „jak Workbench”. To nam da w trybie „ściągającego ekranu” pobranie ustawień z Workbencha – zarówno paleta barw w Workbenchu pozostanie nie rozwalona, jak i będziemy mieć nowoczesny wygląd programu. Z kolei w trybie pracy w Workbenchu konieczne jest (dla uniknięcia np. zostawiania smug po wskaźniku myszy) żeby wejść w Prefs w GUI, i w opcjach GUI dla systemu wyłączyć wsparcie kompozycji obrazu.

Promowanie ekranu jest dobre, ale ma pewien mankament – ustawia

parametry dla programu, który udało się uruchomić. Można wtedy wywołać promotor kombinacją klawiszy i dodać kłopotliwy program do listy (wykrywane są uruchomione aplikacje). Natomiast jeśli wogóle nasz starość nie wystartuje – to jest kłopot, bo nie ma czego wypromować.

Obsługa programów pisanych pod CGX i nietypowych trybów graficznych

Przy okazji starych programów pisanych już pod karty graficzne, trzeba wspomnieć o tym skąd mogą brać się niedogodności. Dawniej były dwa konkurujące

standardy – CGX (CyberGraphX) i Picasso. Ten pierwszy poszedł do MorphOS-a, drugi do nas (w uproszczeniu). **Picasso 96** w AmigaOS 4 ma emulację CGX, ale może się zdarzyć, że nie wszystko zadziała idealnie, ale są to problemy do rozwiązania (opisany przykład ArtEffect 4).

Samo Picasso ze swoim archaicznym panelem przydaje się rzadko, do uzyskania nietypowych rozdzielczości (pod warunkiem, że monitor je obsługuje). Trzeba wpięć włączyć je w tooltype monitora. Być może kiedyś zniknie z systemu, jeśli zostanie rozbudowane Screenmode.

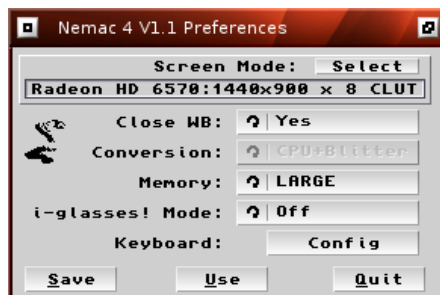
Emulacja poszczególnych chipsetów

Były też inne programy, a w zasadzie „pchelki”, każda do innego zadania.

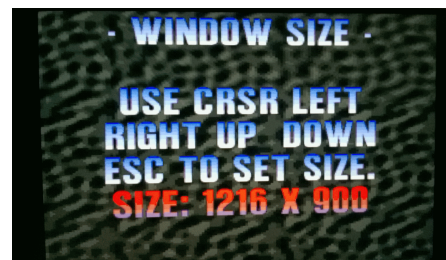
Nalle Puh wyłapywał odwołania do Pauli (układ dźwiękowy obecny we wszystkich modelach Amigi i żadnej AmigaOne). Niestety, raz że nie zawsze (tylko na poziomie systemu operacyjnego, a nie rejestrów sprzętowych); dwa – ten program pod nowym AmigaOS od dawna (trzecia aktualizacja wersji Pre-release AmigaOS 4.0) nie działa.

Był Blitzen – emulator blittera. Blitter to określenie pochodzące od skrótu blit (block image transfer), jego zadaniem było szybkie kopiowanie pamięci wideo, dzięki czemu odciążał procesor.

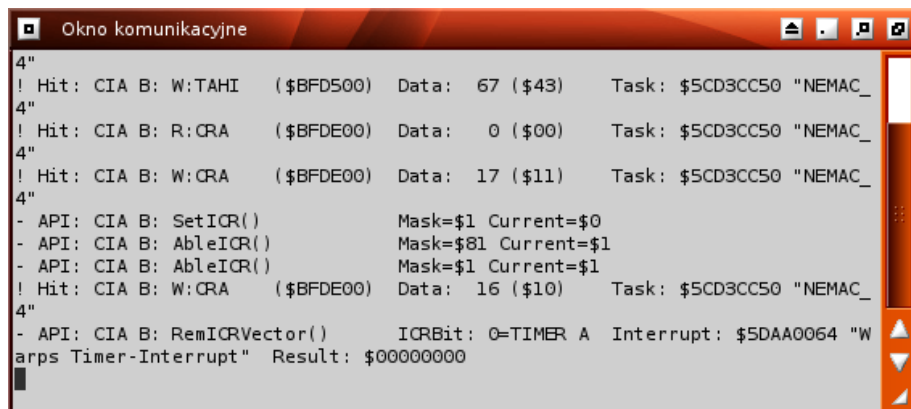
CIAgent z kolei jest emulatorem CIA (complex interface adapters). Wszystkie komputery Amiga (z wyjątkiem CD32, gdzie te funkcje są wykonywane przez układ



Zrzut 9



Zrzut 11



Zrzut 10

Akiko) używają dwóch 8520 CIA, obsługujących port równoległy, klawiaturę, stację dyskiety, przycisk joysticka/myszy.

Współcześnie da się używać tylko CIAgenta (do pobrania na OS4 Depot). Był mi pomocny do poprawnego działania gry „Nemac IV Director's Cut”. Grę wziąłem z serwisu „Amiga Future”, gdzie jest masa legalnego oprogramowania (trzeba być zarejestrowanym użytkownikiem i zalogować się, aby móc pobrać).

Programu nie instalujemy ani nie wrzucamy do C, najlepiej zrobić sobie oddzielny katalog na takie drobiazgi. Najpierw uruchamiamy CIAgenta i po nim kłopotliwą grę. We wspomnianej grze nie korzystamy z promotora – jeżeli Nemac jest na liście w New Mode to trzeba zaznaczyć opcję „do not promote”. To nie wszystko w kwestii ustawień grafiki. Gra ma wstępny konfigurator, ale przy wyborze trybu

graficznego nie możemy wybrać zwyczajowego najlepszego dla swojego monitora (np. Board 0:1920x1080 ARGB32). Trzeba wybrać najlepszy z listy dla swojej karty graficznej (Zrzut 9).

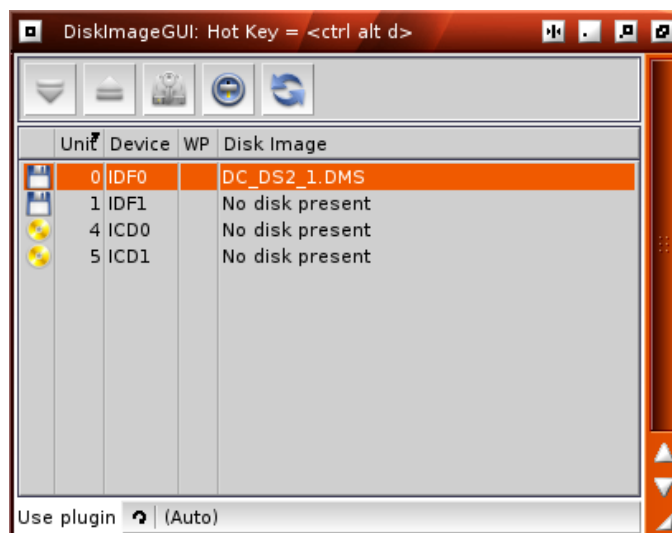
To jeszcze nie gwarantuje sukcesu. Po uruchomieniu nie mogłem przeskoczyć intra i ogólnie klawisze nie działały. Dlatego właśnie niezbędny jest CIAgent. Po uruchomieniu go (przed grą) robi co trzeba. Po zamknięciu gry można zobaczyć w konsoli jego działalność (Zrzut 10).

Zanim doszedłem jak ustawić żeby pograć miałem różne komunikaty: „Can't open screen”, „Can't allocate buffer C2”, „Can't enough Chipmem”. Przy wyłączonym promowaniu, ustawieniach jak na Zrzucie 9 i uruchomionym wcześniej CIAgencie Nemac działa u mnie stabilnie. Jedyne czego mi brakuje to muzyki w intrze.

Wspomnę jeszcze, że po uruchomieniu miałem trochę mniejszą

rozdzielczość niż wybrałem - ekran gry nie wypełniał całego ekranu monitora (*Zrzut 11*). To można swobodnie rozciągać w poziomie i pionie.

Gdyby wyskoczył Grim Reaper (miałem raz taką sytuację), to po ubiciu go gra już była bez dźwięku. Sytuacja ta zdarzyła się przy otwarciu drzwi do sali, gdzie było dużo przeciwników. Gdy powróciłem do ustawień grafiki, które były po pierwszym uruchomieniu domyślnie, problem więcej się nie powtórzył.



(Zrzut 12)

Instalacja programów z dyskietek.

Współczesne modele AmigaOne nie mają nawet kontrolera dyskietek, ale nie jest problemem nośnik a postać, w jakiej stare programy są często dostępne. Mianowicie są to obrazy dyskietek w formacie DMS.

Domyślna akcja po dwukliku w takie archiwum kończy się najczęściej pełnoekranowym obrazem rączki z dyskietką z Workbenchem 1.3. Dzieje się tak dlatego, że obrazy dyskietek domyślnie otwiera emulator RunInUAE. To samo stanie się po kliknięciu w obraz dyskietki ADF (bardziej popularny od DMS, z tym że w olbrzymiej większości ADF-y to są gry). Właściwym postępowaniem w tym przypadku (chęć instalacji starego programu) jest użycie programu **DiskImageGUI**. Znajduje się on w katalogu System. Użycie sprowadza się do zamontowania archiwum *.dms jako wirtualnego napędu. Pojawi się ikonka dyskietki na pulpicie, zawartość można normalnie przeglądać. Dyskietkę montujemy pierwszym przyciskiem, wyciągamy poprzez klik w drugi przycisk (*Zrzut 12*).

UAE

Zasadniczo jestem przeciwnikiem tego rozwiązania. Nie po to zdecydowałem się na Amigę nowej generacji żeby używać emulatora. Mogę pokazać jak zmusić do współpracy stare programy (i o tym jest ten artykuł), ale nie żeby mnie interesowało emulowanie całego komputera. UAE to ja mam na pececie. Dlatego nie przedstawie tutaj konfiguracji, tylko ogólnie napiszę, o co tu chodzi.

UAE zawędrowało na Amigę w czasach betatesterskich AmigaOS 4, kiedy jeszcze równolegle korzystało się z Linuksa (bo przecież AmigaOS 4 dopiero powstawał). Potem zostało przeniesione na AmigaOS 4. Właściwym byłoby operowanie określeniem E-UAE, i to nie jest tak, że to jest jedno rozwiązanie. Jest kilka opcji, zasadniczy wybór sprowadza się do tego, czy chcemy zaemulować cały system czy uruchamiać programy transparentnie.

Najpopularniejsze rozwiązanie to **RunInUAE**. Jest to rzecz o której muszę wspomnieć, ponieważ jest oprogramowaniem dodatkowym

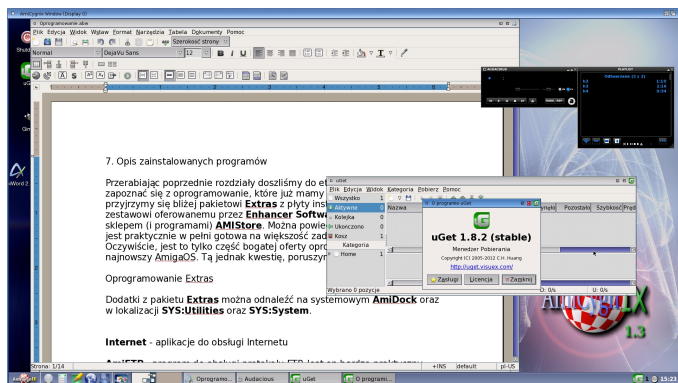
dostarczonym z systemem jako element Extras. Po instalacji zyskamy katalog Emulation, a w nim oprócz emulatora, także Workbench 3.1 oraz ROM-y A1200, CD32 i A500. Dostajemy też możliwość korzystania z gier Cinemaware (do doinstalowania z Extras z płyty z systemem, bądź pobrania ze strony Hyperionu). Są one wstępnie skonfigurowane do pracy z emulatorem, wystarczy zrobić dwuklik na ikonkę. Jest współpraca z WHDLoad, i sensownie to wygląda.

RunInUAE i E-UAE to temat obszerny, na oddzielne opracowanie, i dotyczący głównie gier – a ja chciałem skupić się na programach (Nemac był wyjątkiem dla pokazania użycia CIAgenta).

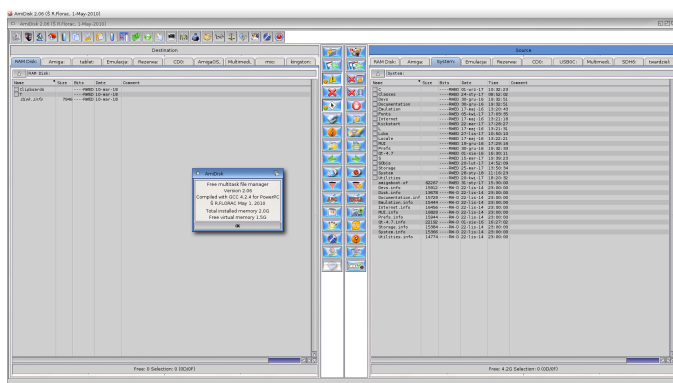
Celem moim było opisanie jak rozwiązano uruchamianie starych programów od strony systemu i jak pokonywać ewentualne trudności nie uciekając się do UAE.

Galeria AmigaOS 4

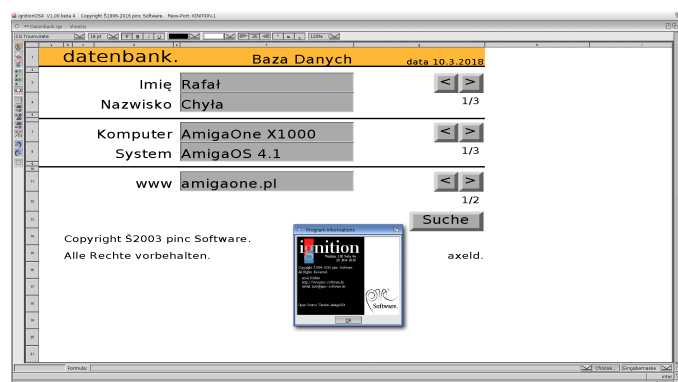
Zrzuty prezentujące AmigaOS 4 na swoich komputerach nadesłali ‘Mufa’ i ‘radzik’.



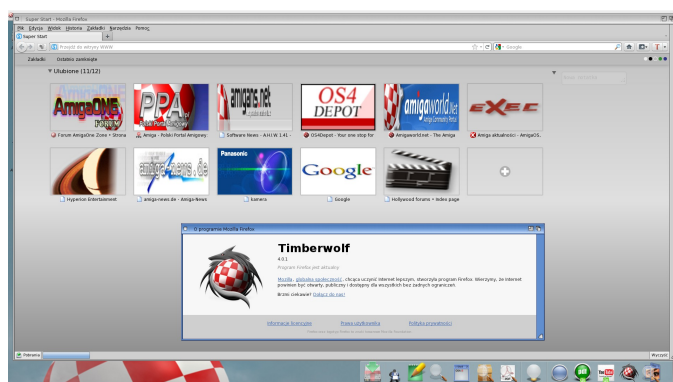
AmiCygrix daje dostęp do wartościowego oprogramowania. Na zrzucie AbiWord, Audacious i uGet.



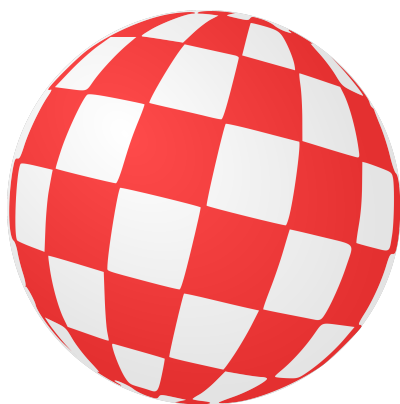
AmiDisk - menadżer plików.



Ignition – arkusz kalkulacyjny.



Timberwolf czyli amigowa wersja Mozilli Firefoxa.



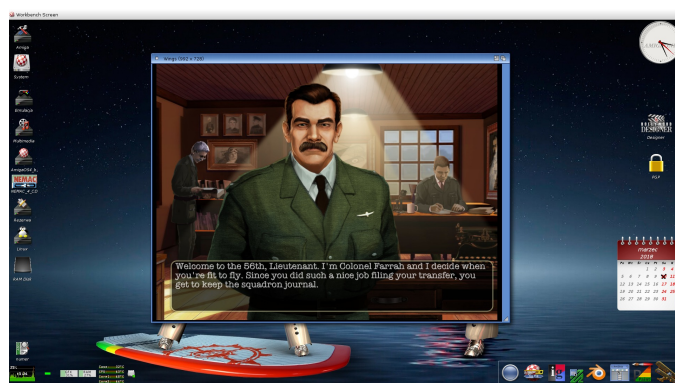
Symbol systemu AmigaOS 4 to Boing Ball – kraciasta białoczerwona piłka, wywodząca się z pierwszego dema Amigi.



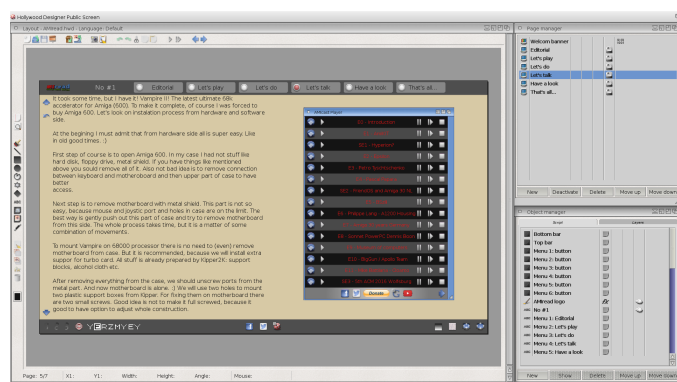
Personal Paint – klasyczny program w najnowszej wersji.



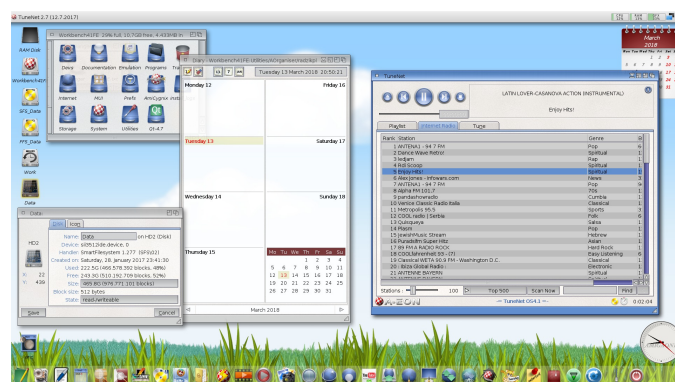
SuperTuxKart – sympatyczna wyścigówka.



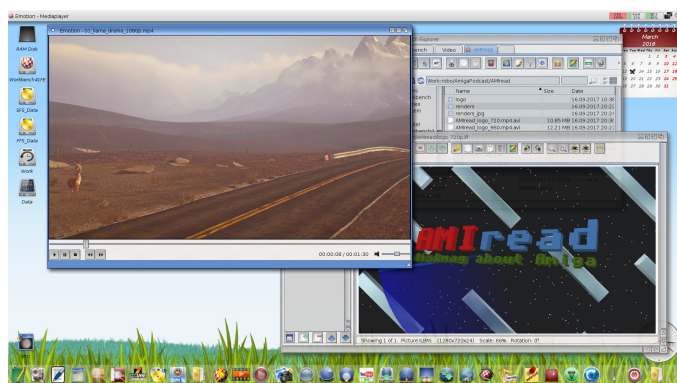
Świetna gra „Wings! – Remastered Amiga Edition”.



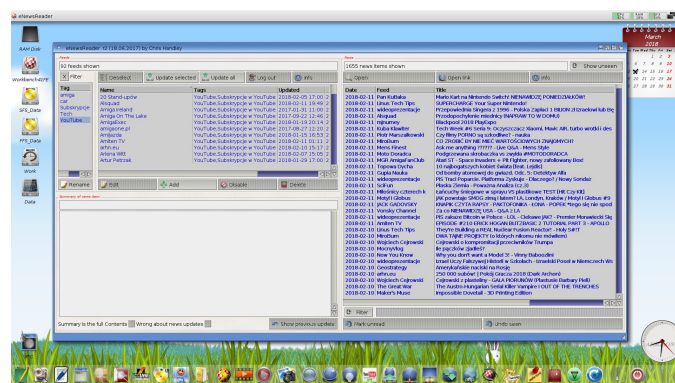
Hollywood Designer – godny następca Scali; można też zrobić w nim interfejs programu, a nawet jak widać diskmagą.



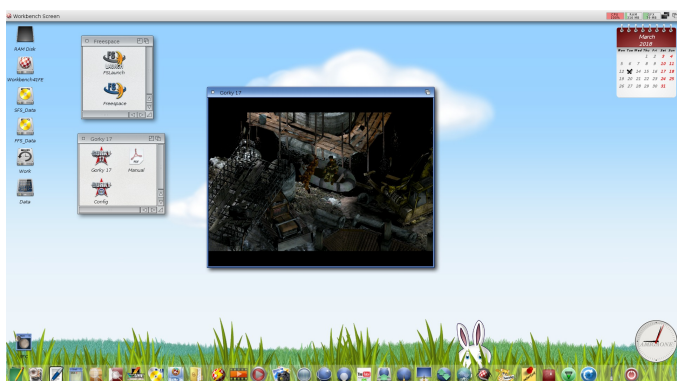
AOrganiser (dziennik z synchronizacją kalendarza Google) i TuneNet (odtwarzacz muzyki i radio internetowe).



Emotion (odtwarzacz filmów) i Workbench Explorer (menadżer plików).



eNewsReader – czytnik RSS.



Gorky 17 - gra.

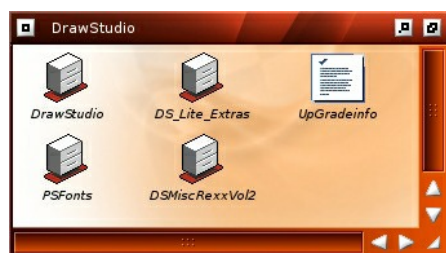


Tower 57 – gra.

DrawStudio

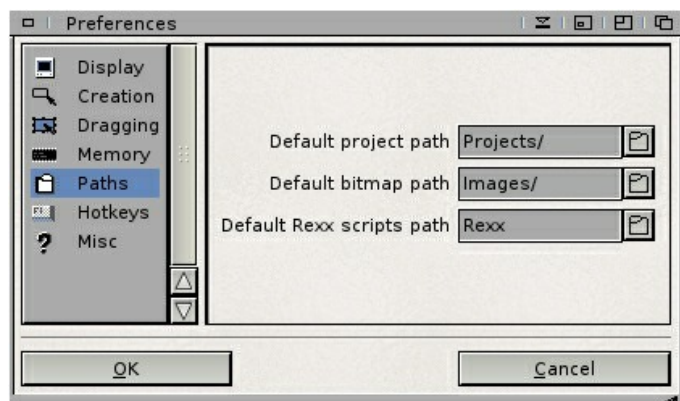
truman burbank

Kontynuujemy temat z poprzedniego numeru, czyli poznawania programów do grafiki wektorowej. Wychodzę z założenia, że macie program już na dysku twardym. Przypominam, że bazujemy na DrawStudio w wersji Lite z „CU Amiga Magazine”.



Zrzut 1

Katalog DrawStudio zawiera w sobie dwa foldery: DrawStudio i DrawStudioFPU. Mają bliźniaczą strukturę katalogów, różnią się plikiem wykonywalnym. Na współczesnych konfiguracjach nie ma to istotnego znaczenia, czy będziemy korzystać z wersji FPU czy zwykłej.



Zrzut 2

DS_Lite_Extras, PSFonts i DSMiscRexxVol2 zawierają różne dodatki, ale przed ćwiczeniami nie będziemy tam zaglądać.

Do wypromowania użyłem New Mode, więcej o tym w tym numerze w artykule o uruchamianiu starszego oprogramowania.

Uruchamiamy DrawStudio.

Na górnej belce ostatnia pozycja Settings-Prefs otwiera okno ustawień. Pierwsze to w zakładce Memory dopisałem po jednym zerze dla zwiększenia limitu pamięci dla fontów i operacji cofnięć.

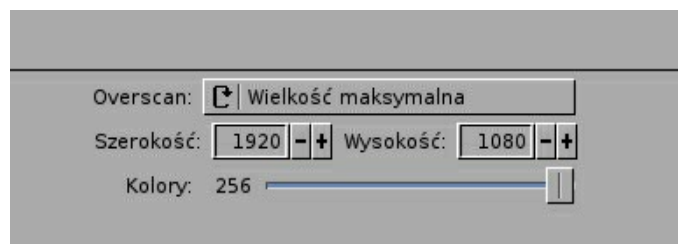
Kolejna pozycja Paths (Zrzut 2) zawiera możliwość zdefiniowania ścieżek dostępu. Domyślnie jest ustawienie tylko dla skryptów (Rexx). Możemy podać ścieżki dla projektów (Default project path) i obrazków (Default bitmap path). Istotne jest Setting-Display (Zrzut 3).

Display type przestawiłem z 1 Bit na 24 Bit. Tam gdzie wchodzi się przez Choose mam Board 0: 1920x1080 ARGB32 i 256 kolorów. Co sobie wybierzemy w Overscan nie ma znaczenia.

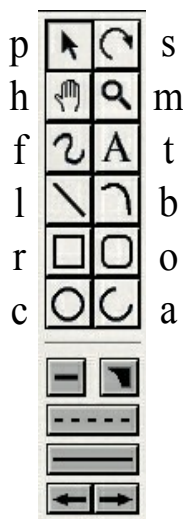
Mimo że mam tak jak na Zrzucie 4, okno wyboru (Zrzut 3) zawsze ma pozycję PAL, ale nie ma co się tym zajmować. Wchodzenie i próby przestawiania są bezzasadne, i skutkują najczęściej przełączeniem się w tryb 16 kolorów. Bierzmy się za program.



Zrzut 3



Zrzut 4



Zrzut 5

Opcje w DrawStudio są łatwe do opanowania, bo nie ma ich wiele. Mamy do czynienia z klasycznym interfejsem – po lewej stronie znajduje się pasek narzędzi, natomiast górne menu zawiera opcje do tych narzędzi, jak i pozostałe funkcje programu.

Pasek narzędzi (Zrzut 5) składa się z dwóch bloków. Opis narzędzia pokazuje się po najechnięciu wskaźnikiem myszy na niego.

Podaję kolejno skrót klawiaturowy, nazwę narzędzia i do czego służy

p – *Pointer* (wskaźnik) – do przesuwania obiektów, zmiany wielkości i edycji
h – *Hand* (ręka) – do przeciągania pracy, chodzi o przesuwanie obszaru roboczego np. gdy pracujemy na powiększeniu (jak przesuwanie kartki ręką); z wciśniętym Shiftem działa szybciej
f – *Freehand* (z wolnej ręki) – do swobodnego rysowania krzywymi
l – *Line* (linia) – do tworzenia prostych linii
r – *Rectangle* (prostokąt) – z wciśniętym Shiftem tworzy kwadrat

c – *Ellipse* (elipsa) – z wciśniętym Shiftem tworzy koło

s – *Rotate* (obróć) – do obracania obiektów

m – *Magnify* (powiększenie) – klik bądź zakreslenie pola daje powiększenie

t – *Text* (tekst) do edycji i tworzenia obiektów tekstowych

b – *Bezier/Polygon* (krzywe Bezie-
ra/wielokąty) – klik i przeciąganie
tworzy obiekt, dwuklik kończy
tworzenie obiektu

o – *Rounded rectangle*
(zaokrąglony prostokąt) – tworzy
prostokąt o zaokrąglonych rogach,
z wciśniętym Shiftem będzie to
kwadrat

a – *Arc* (łuk) – tworzy łuk na bazie
elipsy, przy czym drugi klik ustala
kąt wycięcia

Alter object pen colour – zmiana
koloru pióra obiektu, czyli kolor
obrysu

Alter object fill colour – zmiana
koloru wypełnienia obiektu

Alter object line dash – zmiana
kreski linii obiektu; pozwala
zamienić linie ciągłą na jedną z 8
wersji linii przerywanej

Alter object line thickness –
zmiana grubości linii obiektu

Alter object line end – zmiana
końca linii obiektu (np. na grot
strzałki)

Jeśli chodzi o **górną belkę**, to
składa się ona z zakładek zawiera-
jących kolejne opcje. Są przy
nazwach podane skróty klawiatu-
rowe w kombinacji z klawiszem
A (Amiga), ale już nie podaję,
żeby nie zaciemniać tekstu.

Project

New – tworzenie nowego projektu
Open – otwarcie istniejącego;
domyślne rozszerzenie plików to
*.dsdr

Place – pozwala na umieszczenie
w naszym projekcie innej gotowej
pracy bądź np. obrazka (i do tego
najczęściej używa się); odbywa się
to przez importowanie, czyli dosta-
jemy przymiar którym zakreślamy
pożądany obszar, i importowany
obiekt będzie dokładnie tyle zaj-
mował

Save – zapis poprzez nadpisanie
aktualnie otwartej pracy

Save as – zapis pod nową nazwą

Close – zamyka dokument

Print – drukowanie

Export-Bitmap – do zapisu doku-
mentu w formacie obrazka; przy-
kładowo możemy wyeksportować
wszystkie obiekty, cały dokument
czy tylko zaznaczony obiekt; moż-
liwości zapisu do formatów: BMP,
GIF, IFF-ILBM, JPEG, PCX i
TIFF

About – metryczka programu

Info – informacje na temat doku-
mentu (liczba operacji możliwych
do cofnięcia, ilość obiektów, ile
zajmuje pamięci, ile jest wolnej
pamięci w komputerze i podobne)

Close – zamyka program

Edit

Undo – cofnij

Redo – ponów

Cut – wytnij

Copy – skopiuj

Paste – wklej

Delete – wymaż (skasuj)

Clone – sklonuj

Duplicate – stwórz kopię

Select all – zaznacz wszystko

Tu chcę wspomnieć o tym, że klon
i duplikat powinny działać różnie.
Klon powinien uwzględniać
wszystkie zmiany w oryginale – co
zmienimy w oryginale, to zmienia
się i w klonie. Natomiast duplikat
jest tylko kopią, będącą oddziel-
nym bytem. U mnie nie działa to,

funkcje te dają ten sam efekt (kopia obiektu).

Jak łatwo zauważyć, same dwie pierwsze pozycje to już jest 20 opcji – musiałbym napisać książeczkę będącą instrukcją obsługi, wręcz podręcznik programu, dlatego podam tylko z grubsza o co chodzi.

Wszystkie polecenia są oparte na prostym angielskim, i w przyszłości menu nie będzie tak rozpisywane. Jeżeli ktoś nie zna języka – proszę przyswoić powyższe słówka, bo to są podstawowe opcje powtarzające się w niemal wszystkich programach.

Czas coś podzielać.

Na tę chwilę interesuje nas katalog *Projects* w folderze programu. Będą tam trzy projekty (samochód Formuły 1, scyzoryk i napisy). Wybierzmy napisy – plik to *TextEffects.dsdr* (Zrzut 6).

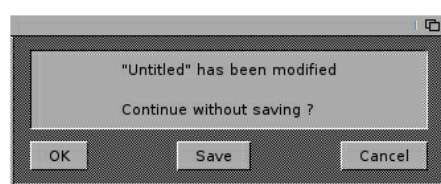


Zrzut 6

Niestety wersja Lite ma pewne ograniczenia, i o ile nie dotyczą one narzędzi, o tyle zauważymy je w wyeksportowanych plikach (brak obsługi głębi kolorów 24

bit). Niemniej powyższy obrazek demonstruje część możliwości. Napisy pokazują użyte funkcje: Gradient – użycie gradientu, Warp! – wykręcanie napisu, Transparent – przezroczystość, Shadow – rzucanie cienia i Fade – zanikanie.

Istotna informacja niezbędna w dalszej pracy – przy zamykaniu projektów wyskakuje taka tabliczka:



Zrzut 7

Klik w *OK* spowoduje zamknięcie pracy bez wprowadzania zmian (jeśli to był nowy projekt – wszystko tracimy), *Save* to zapisanie naszych działań, a *Cancel* to powrót do pracy.

Otwieramy nowy projekt i wybieramy nadrzędzie *Text*. Następnie z górnej belki *Text-Size-48 pkt*. Lepiej wybrać duży napis, łatwiej widać na nim użyte efekty. I piszemy cokolwiek. Następnie wybieramy narzędzie *Pointer* i klikamy w napis. Obiekt jest aktywny, teraz wybieramy z górnej belki opcję *Text-Font* i tam w okienku możemy zmienić krój czcionki na inny (Zrzut 8).

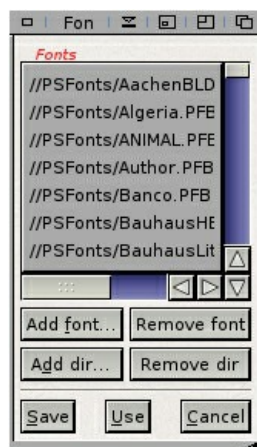


Zrzut 8

Następnie wybieramy (na pasku narzędzi) kolor wypełnienia napisu i obrysu liter. Jeżeli na tym etapie coś nie działa (np. nie zmienia się kolor obrysu mimo tego, że go wybraliśmy) trzeba wybrać z górnej belki polecenie *Object-Convert to-Bezier*. Z tym że po tej operacji zostaną utracone właściwości czcionki (napis zamieni się w krzywe wektorowe), co w praktyce oznacza brak możliwości zmiany kroju. Nie będą również działały inne ustawienia dla fontów np. zmiana wielkości, ale z tym łatwo sobie poradzić – można napis rozciągnąć zwyczajnie myszką, z wciśniętym klawiszem Shift aby rozciągało go proporcjonalnie. Czyli efekty w napisie stosujemy wtedy, gdy mamy pewność, że nie będziemy go zmieniać. Tak naprawdę to żadne utrudnienie, bo „zrobienie napisu” to

przecież napisanie ledwie wyrazu. No ale wspominam o tym żeby wiedzieć.

Żeby mieć większy wybór fontów trzeba wybrać z górnej belki *Text-Font Manager* i w okienku jakie pokaże się wskazać poprzez klik w *Add dir* ścieżkę do katalogu *PDFonts* (Zrzut 9).



Zrzut 9

Jeżeli chodzi o dodawanie efektów to najbardziej istotne są w zakładce *Effects-Rexx*, przy czym nie do końca działają tak jak chcielibyśmy. Na przykład cień jest od razu oddzielnym bytem, a użycie funkcji *Undo* nie spowoduje zniknięcia cienia lecz jego modyfikację. Efekty te są przeważnie nie tyle zmienianiem tekstu, co służą do urozmaicenia napisów, takie obiekty towarzyszące.

Ćwiczenie 2

Cel: modyfikacja parametrów w napisach, tworzenie własnych wypełnień

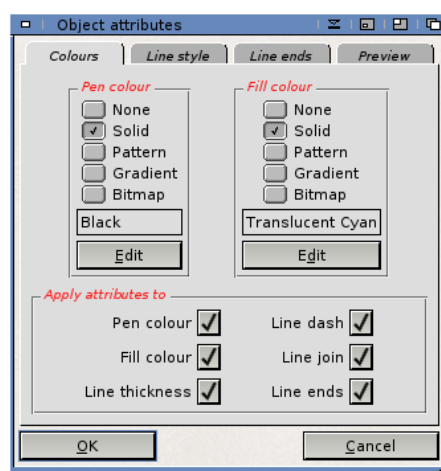
Powyższe ćwiczenie to było takie wstępne. Właściwe tworzenie napisów odbywa się przy pomocy innego narzędzia.

Otwórz ponownie projekt *TextEffects.dsdr*. Użyjemy go do mody-



Zrzut 10

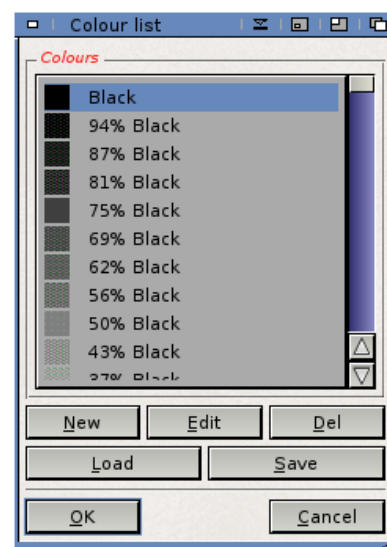
fikacji ustawionych parametrów w napisach, aby poznać nowe opcje. Po otwarciu projektu sięgnij do górnej belki i wybierz *Object-Attributes*. Pokaże się zakładka *Colours* (Zrzut 11) mająca odpowiednio ustawienia dla *Pen* (obrys) i dla *Fill* (wypełnienie).



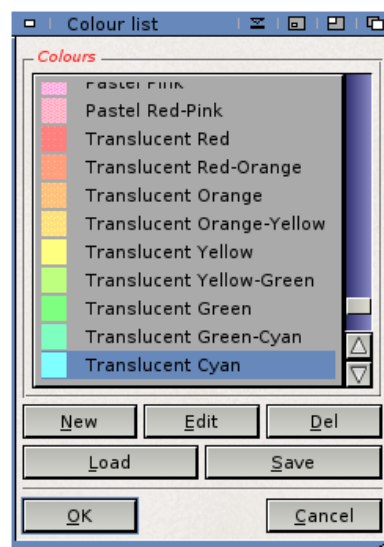
Zrzut 11

Klik w przyciski *Edit* otwiera listy (Zrzuty 12a i 12b), ale to są części tej samej listy.

Kolejny klik w *Edit* zaprowadzi nas do okna *Edit colour* (Zrzut 13). Ale to wszystko dzieje się jeśli zmieniamy ustawienia dla rodzaju wypełnienia *Solid* (jednolite). Jak popatrzymy na Zrzut 11 to mamy



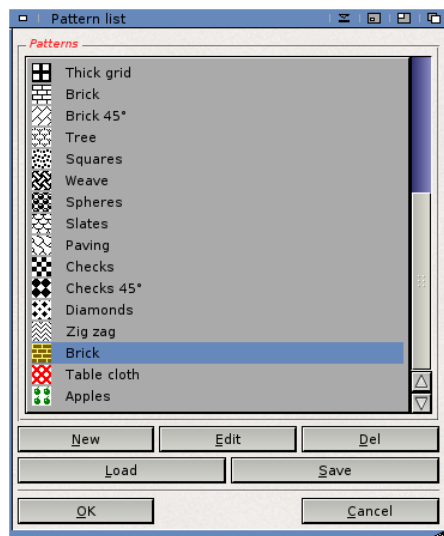
Zrzut 12a



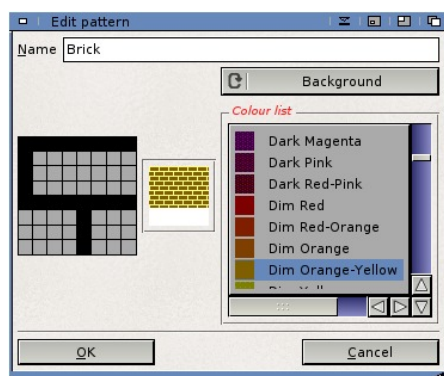
Zrzut 12b

jeszcze *None*, *Pattern*, *Gradient* i *Bitmap*.

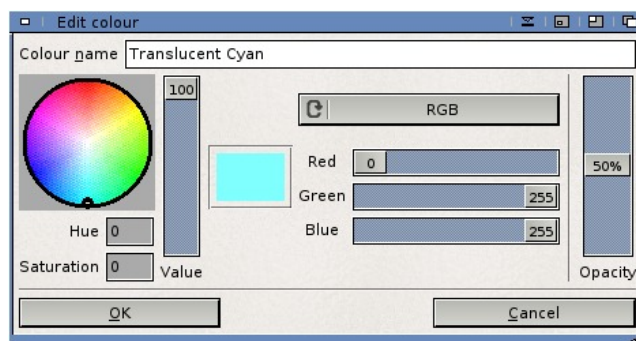
None jest przydatne jeśli chcemy mieć napis bez obrysu bądź pusty. *Pattern* ma swoją listę wyboru (Zrzut 14a) i inny edytor (Zrzut 14b).



Zrzut 14a

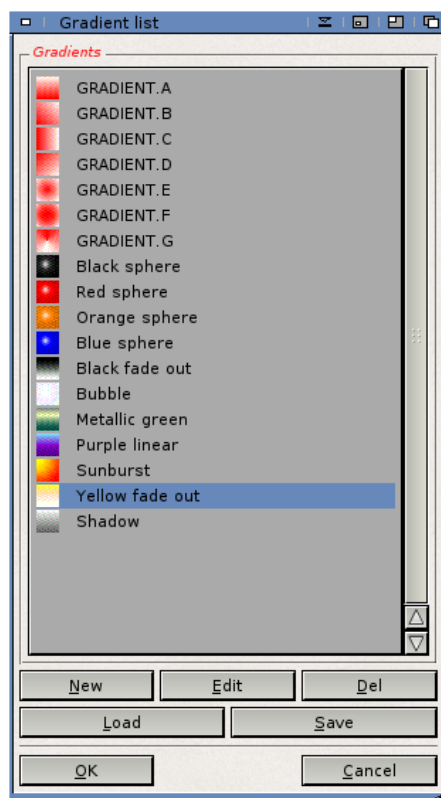


Zrzut 14b

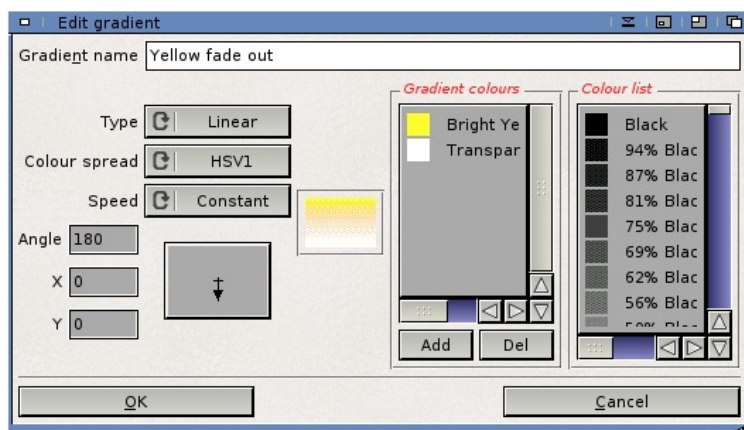


Zrzut 13

Gradient również (Zrzuty 15a i 15b).



Zrzut 15a



Zrzut 15b

Może się zdarzyć że przy użyciu *Load* zobaczymy puste okno, najczęściej w sytuacji gdy zmienimy kategorię wypełnienia (np. z *Pattern* na *Solid*). Trzeba wtedy wskazać plik z listą. Będzie monit jak na obrazku (Zrzut 16).



Zrzut 16

Ja przy wyborze gradientu klikałem *No*, tym samym nie usuwając bieżącego otwartego pakietu, i otwierałem kolejne (są w *DS_Lite-Extras/Gradients*). Zbudowałem w ten sposób „superlistę” zawierającą w sobie wszystkie dostarczone z programem gradienty (zapisalem ją oczywiście). Mi tak było wygodnie, ale ktoś inny może skorzystać z tego, że można tworzyć własne, i chcieć mieć listy tematyczne (np. oddzielna lista na przezroczystości). Listy muszą być oddzielne dla różnych typów wypełnień.

Przykładowe efekty edycji ustawień wczytanego projektu pokazuje Zrzut 17 (porównajcie ze Zrzutem 6).

Bardzo dużo miejsca poświęciłem robieniu napisów, ale są ważne powody: duży wybór funkcji, i najmniej stresujące w porównaniu z używaniem innych narzędzi w DrawStudio.

Ćwiczenie 3 (Zrzut 18)

Cel: wstawianie obrazków, nakładanie napisu

Nadszedł czas zrobić użytek z poznanych możliwości programu. Obrazki wstawiamy opcją *Project-Place*. Jeżeli chcemy wyrównać wybrane elementy względem siebie, można to robić strzałkami na klawiaturze. Z wciśniętym klawiszem *Shift* przesuwanie jest szybsze. Jeżeli obrazki nie dadzą się spasować, można je układać „na zakładkę”, i potem za pomocą opcji z belki w *Object-Arrange-...* ustalić ich kolejność.

Użyłem obrazków z zasobów programu, ale nic nie stoi na przeszkodzie żeby tą metodą zrobić własną pocztówkę.

Ćwiczenie 4 (Zrzut 19)

Cel: nauka tworzenia własnych obiektów.

Pierwszy zonk jaki zaliczyłem to: jak uzyskać figury inne niż na pasku narzędzi, choćby zwykły trójkąt? Spróbujmy.

Na tym etapie poznawania programu miałem chwile zwątpienia. Wspomniany trójkąt zrobiłem przez odjęcie wężła z prostokąta. To co widać na obrazku to są modyfikacje początkowego prostokąta widocznego w lewym dolnym rogu zbioru figur.

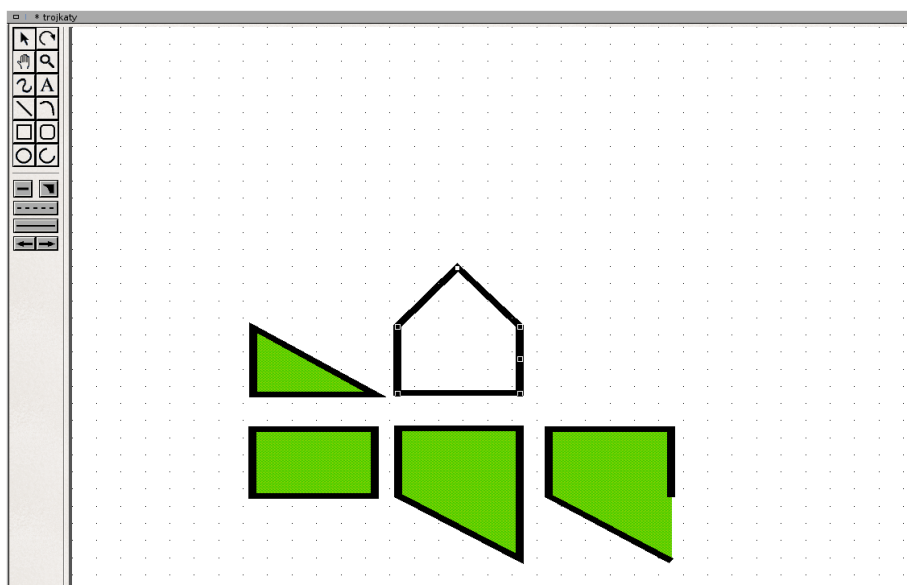
Wyjaśnienia wymaga jak powstawały poszczególne obiekty. To co wygląda jak koperta jest to tryb edycji węzłów, dlatego nie widać koloru tylko same odcinki.



Zrzut 17



Zrzut 18



Zrzut 19

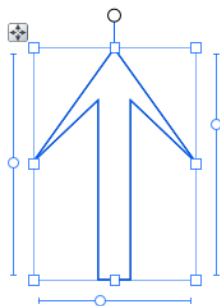
Narzędzia są na belce w sekcji *Bezier*:

Edit Points, *Add Points*, *Delete Points* – narzędzia edycji węzłów, można je przesuwać, dodawać i usuwać. Tak tworzymy nowe kształty – węzły dają się przeciągać, i im jest ich więcej, tym bardziej skomplikowane kształty możemy tworzyć.

Make Open i *Make Closed* służą do otwierania i zamykania obiektu. Widać to w dolnym rzędzie na ostatnich dwóch prostokątach, w których w trybie edycji węzła rozciągałem jeden bok figury. Jak widać przy obiekcie otwartym wypełnienie jest, ale ono w miejscu niedomknięcia nie ma krawędzi.

Opcje *Make compound object* i *Split compound object* służą odpowiednio do scalania i rozdzielania obiektów. Jest to przydatne na przykład wtedy, gdy rysujemy odcinkami, i potem chcemy, żeby po złożeniu dawały się przesuwać jako całe obiekty (np. w diagramach). Nie stosujemy tego do scalania pełnych obiektów, tylko tych ze stawianych kresek.

Gwoli uzupełnienia, we współczesnych programach są opcje umożliwiające przerabianie gotowców, i jest to zrobione niesamowicie wygodnie. *Zrzut 20* pokazuje jak rozwiązane jest to w windowso-
wym Serif DrawPlus X8.



Zrzut 20

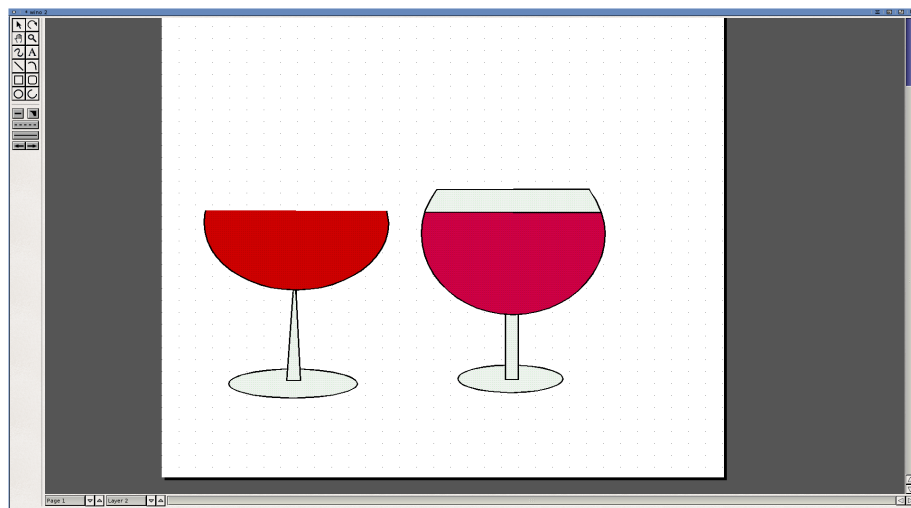
No ale nam zachciało się starszkołnego dłubania, więc brniemy dalej.

Ćwiczenie 5 (Zrzut 21)

Cel: pierwsza własna praca w wektorach

Pierwszy kieliszek to trzy elementy. Stopa zrobiona jest z elipsy, nóżka to trójkąt zrobiony z wąskiego, pionowego prostokąta, a czasza powstała przy użyciu narzędzia *Arc*. To prymitywna wprawka, która ma was ośmielić. Drugi kieliszek wymaga wyjaśnienia jak powstała czasza. To są dwie elipsy, do których w górnej części dodałem węzły, a potem najwyższy usunąłem, dzięki czemu te elipsy „ściąłem”. Żeby ładnie pasowało zrobiłem to w ten sposób, że po ścięciu pierwszej elipsy sklonowałem ją i przyciąłem kłona drugi raz, dzięki czemu krzywizna obu elementów po złożeniu jest idealna.

Czas na ostatnie ćwiczenie, łatwiejsze i przyjemniejsze. Mi przynajmniej to co pokażę otworzyło oczy na program (w sensie „przestał mnie uwierać”).



Zrzut 21

Ćwiczenie 6 (Zrzut 22)

Cel: tworzenie własnych kształtów „z ręki”

Nie pokazywałem rysowania. Były kształty ale nie było typowego stawiania kresek. Tak jak koło i okrąg różnią się (okrąg jest pusty), tak i jest w programie. Niestety nie wykombinowałem jak zalewać takie puste figury, ale musi być jakiś sposób. Przecież byłoby to niedorzecznością jakby się nie dało. Tworzenie kolejnych obiektów poprzez przerabianie istniejących prymitywnych możliwości typu koło i kwadrat to byłaby jakaś masakra.

Nie wiem, może da się to zrobić w cywilizowany sposób, ja sobie poradziłem poprzez rysowanie z ręki.

Na zrzucie widać narysowanego jakiegoś głupka, nie jest to istotne, opisuję zasady. Oczy robimy z elips. Po zrobieniu jednego oka i pokolorowaniu go, łączymy elementy używając opcji *Object-Group*. Następnie *Edit-Clone*, dzięki czemu oczy będą jednokowe. Pisałem wcześniej, że *Clone* i *Duplicate* dają ten sam efekt. To prawda, ale nie do końca, jest pewien detal. Przy wybraniu *Clone*

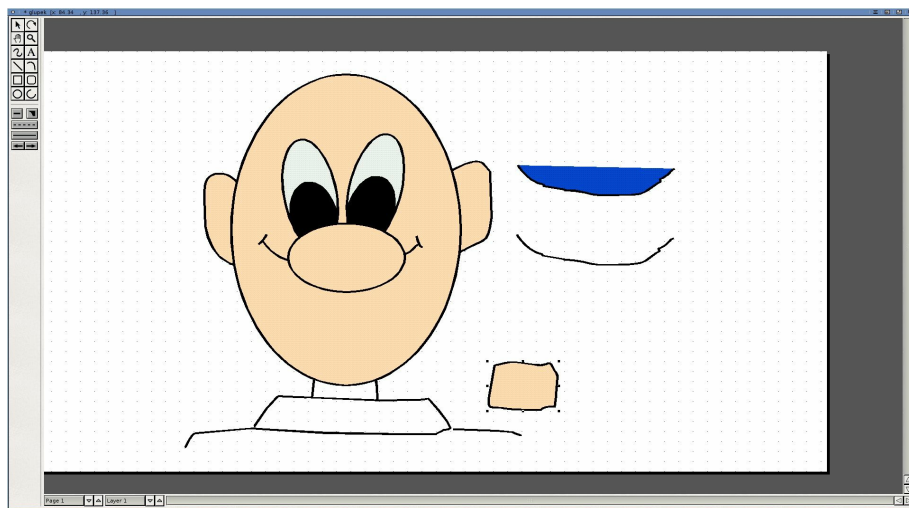
tworzona jest kopia z małym przesunięciem względem oryginału, a przy *Duplicate* otwiera się okno dialogowe w którym można zmienić pewne parametry, ale domyślnie kopia tworzona jest w tym samym miejscu co oryginał (na nim).

Oczy możemy ustawić narzędziem *Rotate*. Uszy narysowałem narzędziem *Freehand* i przesunąłem na spód.

Problemem okazał się uśmiech. I tu wyszła fajna rzecz w DrawStudio – kształty rysowane z ręki zalewane są automatycznie. Nie trzeba robić dwukliku na ostatnim węźle jak w wielu programach, choćby nie wiem jak nabazgrać to program to domknie. No tak, ale tym sposobem nigdy nie narysowalibyśmy odręcznej krzywej, prawda? Rozwiązanie okazało się proste – trzeba wejść w atrybuty obiektu (*Zrzut 11*) i przestawić wypełnienie na *None*. Po prawej stronie „twarzy” widać ten sam „uśmiech” z wypełnieniem i bez niego, to jest ten sam obiekt. Jest to niestarczanie zrobione, ale po wsunięciu pod nochal nie widać mankamentów. Kąciki ust to zwykłe odcinki.

Szyja i sweterek były szkicowane. I znów – ten spłachetek obok to pokazuję że można, nawet niestarczanie, narysować szyję po to by ją wsunąć pomiędzy inne obiekty. No bo nie odkryłem (a może nie da się) jak wypełniać puste obiekty.

Cóż, dopiero w tym miejscu zaczyna się „mięsko”, a tu trzeba kończyć. Nie opisałem wszystkiego (np. jak kłaść napisy wzdłuż krzywych, czy jak kreatywnie wykorzystać duplikowanie). Podam kilka rzeczy już skrótowo, resztę musicie odkryć sami (a do



Zrzut 22

odkrycia jest sporo, z tego co widzę po *DS_Lite_Extras*).

W lewym dolnym rogu jest coś takiego jak *Page* (strona) i *Layer* (warstwa). Można tworzyć nowe i przełączać się między nimi. Warstwy działają tak, że widoczne są wszystkie, ale naraz wybrana jest tylko jedna. Pozostałe są niewidoczne, więc pracując nic nie zepsujemy na innych warstwach. Rysujcie grubymi liniami, lepiej widać wtedy wszelkie nierówności odcinków; grubość można zmniejszyć jednym kliknięciem, gdy obiekt jest gotowy.

Do rysowania prostych kształtów warto używać klawisza *Shift*, dzięki czemu odcinki są kładzione równo.

Fajną cechą jest możliwość zaznaczenia kilku obiektów (w przypadku wyżej np. zakreszenia całości) i jednym kliknięciem zmiana grubości wszystkich linii naraz.

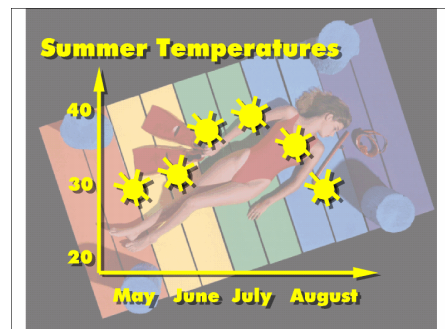
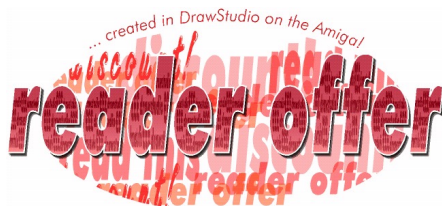
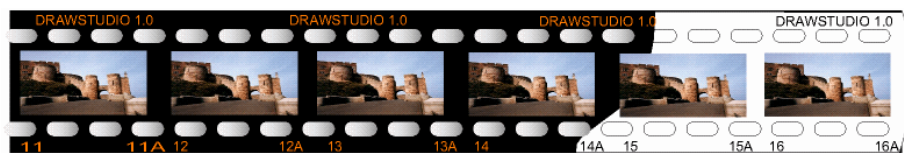
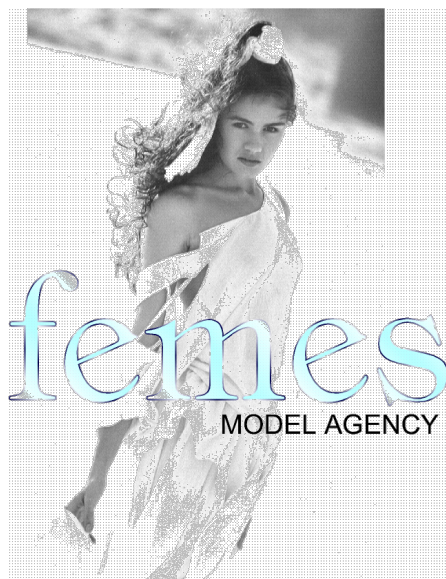
Istnieje spolszczenie do DrawStudio, do pobrania z Aminetu, ale obecnie nie wyświetla poprawnie polskich znaków; wymagana jest konwersja z AmigaPL na ISO.

DrawStudio jest stabilny. Dwa razy mi „grimował”, ale to nie wywalalo programu – zjechałem ekranem w dół, zamknąłem tabliczkę pod spodem, i dłuhałem w programie dalej. W razie kłopotów z odświeżaniem obiektów użycie narzędzia *Hand*.

W zeszłym numerze czasopisma obiecałem szkółki DrawStudio i AmiFIG-a. Niestety wychodzi to obszernie, a nie ma sensu na siłę skracać tekstów, bo wtedy nie spełnią swojej roli. Nie chciałem dzielić tematu na odcinki, bo ukazujemy się co kwartał.

Udało się omówić konfigurację, większość narzędzi, i zamieścić kilka ćwiczeń. Nie opierałem się na żadnych materiałach, być może to pierwsza szkółka DrawStudio na świecie, po 22 latach od wydania. Program odkrywałem sam, idąc od zera, więc pewnie można było opisać go lepiej.

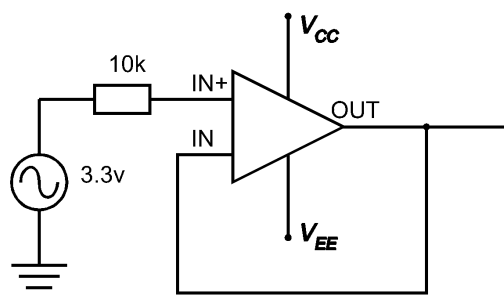
Mam w planie jeszcze odkurzyć kilka takich wykopalisk, ale to wszystko dopiero się okaże. To muszą być programy, które działają w miarę stabilnie pod nowym AmigaOS.



Podejdźcie do DrawStudio z wyrozumiałością i nie zniechęcajcie się. Dołączone do niego jest wiele dodatkowych projektów, ale nie wszystkie są wyeksportowane do IFF, dlatego warto przeglądać oryginalne pliki *.dsdr. Wiedziecie, że niektóre mają wiele stron, a nieliczne przydatne adnotacje. Zamieszczam niektóre z nich, powinno was to zachęcić.

Kiedyś trzeba postawić kropkę, więc – powodzenia.

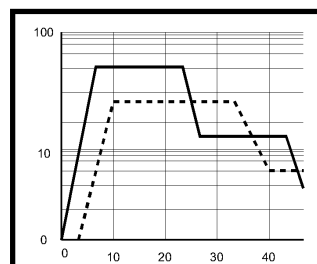
Voltage follower configuration for slewrate comparison



```

UA741 SLEW RATE
* MACROMODEL
X1 1 3 3 4 5 UA741MAC
* FULL MODEL
X2 2 6 6 4 5 UA741MAC
VIN1 10 0 DC .275M PULSE 5
VIN2 11 0 DC .432M PULSE 2
RIN1 10 1 100
RIN2 11 2 100
VCC 4 0 15
VEE 5 0 15
*
.TRAN .25U 50U
.OPTIONS ACCT
.END

```



Hollywood #3

Czas

ferin

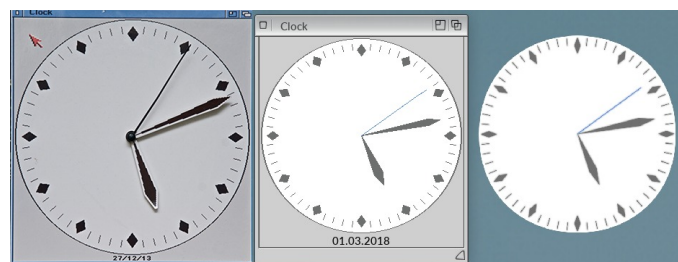
Dziwna rzecz z tym czasem, niby jest a ciągle go brak. Niektórzy mówią, że wcale nie istnieje, że jest tylko abstrakcyjną wielkością umożliwiającą obliczenia fizyczne, coś jak zero w matematyce, które jest zerem, czyli brakiem czegoś, a jednak przewija się na każdym kroku.

Inni znowu twierdzą, że jest tylko tu i teraz, bez przyszłości ani przeszłości. Ale ci najdziwniejsi mówią, że nie tylko przeszłość ma wpływ na przyszłość ale także, że to co się jeszcze nie wydarzyło na to co już się stało...

Niestety nie rozstrzygniemy tego tutaj, możemy natomiast mierzyć to co wszyscy nazywamy upływem czasu, bez względu na to czy istnieje czy nie. W związku z tym napiszemy sobie skrypt, który będzie nam rysował wektorowy zegar, będzie to lepsza wersja systemowego Clocka.

Ktoś może zapytać – po kiego skoro już jest? Do głowy przychodzi mi co najmniej kilka powodów. Po pierwsze, będę mógł sobie go odpalić na którymkolwiek systemie, czy windzie, czy mac osie, a nawet andku i na iosie. Po drugie dodam kilka rzeczy których nie ma oryginał. Poza tym czuję jakiś dziwny sentyment do tego „brzydala”. Kiedyś nawet zrobiłem sobie wersję „namacalną” i od tej pory zawsze mierzy mi upływ czasu stojąc na półce powyżej.

W tej części będzie sporo nowych rzeczy, ale najpierw chciałem podziękować teh_KaiNowi za pomoc,



Zrzut 1

bo bez niego by się nie udało. Dotarcie tu gdzie jestem z budową tego zegara to była ciężka przeprawa, bo gdy już myślałem że mam wszystko i będzie już z górki to zawsze pojawiała się jakaś niby pierdoła, którą musiałem jakoś poprawić albo obejść, używając swoich (wciąż nader skromnych) umiejętności w pisaniu skryptów Hollywoodowych. No ale jakoś się udało, uff.

Niestety ze względu na to, że „Amiga NG” będzie teraz drukowana, to w tej części skupię się jedynie na stworzeniu samego zegara, bez dodatków, które sobie wymyśliłem. Jedyną nowością będzie to, że zegar nie będzie miał tła w postaci okienka, nie będzie tam nic. Chciałem sprawdzić czy możliwe w Hollywood zrobić taki zegarek, jakie choćby są w pakiecie Enhancera dla AmigaOS 4.1.

No dobra, żeby zrobić takie przeźroczyste tło, musimy zastosować pewien trik, którego nauczył mnie jPV – autor różnych programów zaczynających się na RNO -comics, -anim, Desktop widgets i pewnie wielu innych. Ponieważ Hollywood potrzebuje tła żeby móc cokolwiek wyświetlić i określić rozmiar okna, musimy sobie stworzyć najpierw tarczę zegara, która będzie naszym tłem i zarazem określi jego rozmiary, dzięki temu możemy pominąć atrybuty width i height komendy preprocesora @DISPLAY.

Nie będę tutaj opisywał skryptu linia po linii, bo część rzeczy jest już Wam znana, a poza tym starałem się dokładnie wszystko skomentować w kodzie; opiszę pokrótce nowe funkcje i (ich) atrybuty.

Nowe atrybuty funkcji preprocesora @DISPLAY

Hidden = True

Powoduje że okno naszego programu nie pojawi się do pólki nie użyjemy funkcji **OpenDisplay(ID)**. Przyczyną dla której używamy Hidden jest fakt, że

najpierw nasz program sam musi stworzyć swoje tło i dopiero wtedy będziemy mieli co pokazywać.

NoModeSwitch = True uniemożliwia przełączanie programu z trybu okienkowego na pełnoekranowy. W końcu ma to być zegarek na blacie WB.

ScaleMode = #SCALEMODE_AUTO

Automatyczny wybór sposobu skalowania zawartości okna, drugą opcją która mamy to **#SCALEMODE_LAYER**. Podobno daje lepszą jakość skalowania, aczkolwiek nie jestem w stanie tego stwierdzić, bo gdy użyłem tej opcji to przy zmianie wielkości okna następuje zawieszenie systemu. Podobnie wygląda sytuacja gdy pominiemy ten atrybut, bo **SCALEMODE_AUTO** to ustawienie domyślne, więc można by go pominąć.

Jest jeszcze opcja **#SCALEMODE_NONE**, którą ustawiamy gdy nie chcemy skalować naszego okna.

Sizeable = True

Włącza skalowanie okna, domyślnie możemy uchwycić dolny prawy róg, podobnie jak w okienkach systemowych. Ale w naszym wypadku to niemożliwe, bo okienko naszego zegara jest w pełni przezroczyste, co powoduje, że nie możemy go kliknąć.

SizeRegion

Na początku jedna uwaga, oszczędzi Wam rwania włosów z głowy, jak było w moim wypadku, a mianowicie atrybut ten oczekuje **listy tabeli**, nawet jeśli chcemy mieć tylko jeden obszar za który możemy przeciągać nasze okno, musimy zamknąć nasze tagi podwójnymi **nawiasami klamrowymi** `{ }`.

Atrybut ten oczekuje następujących wartości: **SizeRegion={ {Type, X, Y, Width, Height} }**

jako **Type** podajemy **#BOX**, jedyny akceptowany rodzaj, nie ma tu żadnego wyboru. **X** i **Y** to współrzędne gdzie ma się zacząć obszar rozciągania, a **Width** i **Height** to jego szerokość i wysokość.

To wszystkie nowe rzeczy jeśli chodzi o preprocesor.

Następna funkcja to **EscapeQuit(True)**, powoduje zakończenie działania programu przez naciśnięcie klawisza **ESCAPE**.

Zanim przejdę dalej, wspomnę tylko, że jeśli chcecie możecie zassać kolorowanie składni dla edytora

Annotate, plik jest dostępny w tym samym miejscu gdzie reszta plików szkółki.

Następny element skryptu to zmienne globalne, które zadeklarowałem żeby móc ich używać w kilku miejscach, jednocześnie nadając im nazwy które będą znaczące. Ogólnie nie zalecane jest używanie zmiennych globalnych, lepiej używać lokalnych, bo są „łżejsze”, ale nasz skrypt nie będzie bardzo wymagający więc tym razem możemy sobie na nie pozwolić.

z_Radius = 225

z_CX = 250

z_CY = 250

Pierwsza zmienna określa odległość punktów minut i godzin od środka zegara, dwie pozostałe wyznaczają jego środek. Przedrostki „z_” - jak zmienna - wymyśliłem sobie sam żeby łatwiej mi było odnajdywać je w kodzie; żeby dodatkowo sobie ułatwić życie, dodałem ich kolorowanie (*Zrzut 2*).

Teraz w końcu narysujemy tarczę zegara. Jak już wspomniałem, zegar będzie wektorowy, nie użyjemy żadnych plików zewnętrznych z grafiką, wszystko stworzymy w skrypcie.

Żeby móc stworzyć jakikolwiek obiekt wektorowy musimy zdefiniować najpierw rodzaj jego wypełnienia (albo jego brak, czyli same linie krawędzi), możemy też określić czy chcemy wygładzanie, albo by nasze figury rzucały cień. Służą do tego funkcje z biblioteki **DRAW** (Draw library).

SetFillStyle(#FILLCOLOR) powoduje użycie koloru do wypełnienia, nie definiuje jaki ma to być kolor i czy ma być przezroczysty. To określamy w innym miejscu.

SetFillStyle(#FILLNONE, 1) powoduje rysowanie jedynie krawędzi wektorów linią o grubości jednego piksela. Możemy jeszcze użyć **#FILLGRADIENT** i **#FILLTEXTURE**, ale nie będę ich tu opisywał bo nie będą potrzebne, możecie sami poczytać w manualu.

SetFormStyle(#ANTIALIAS) wygładza krawędzie, do wyboru mamy jeszcze:

SetFormStyle(#SHADOW) - cienie,

SetFormStyle(#EDGE) - dodaje obwódkę wokół wektora


```

@VERSION 7,0
@DISPLAY {
    Borderless = True,           ;okno bez ramki
    Hidden = True,               ;"ukrywam" okno bo nie chcę żeby było widoczne aż do czasu
                                ;stworzenia tła wektorowego
    NoModeSwitch=True,          ;nie chcę żeby użytkownik zmieniał wyświetlanie na pełen ekran
    Layers = True,               ;włączam warstwy
    DragRegion =                 ;ustalam obszar za który będę mógł przesuwac zegar,
                                ;muszę pamiętać, że części ekranu z pełną przeźroczystością
                                ;są "nieuchwytnie", innymi słowy mogą targać zegar tylko za
                                ;to co widać, w tym wypadku za okrągłą tarczę zegara, pomimo
                                ;że zdefiniowany niżej obszar ma kształt kwadratowy.

    {
        {Type = #BOX, X=0, Y=0, Width=500, Height=500}
    },
    ScaleMode = #SCALEMODE_AUTO, ;tryb skalowania zawartości okna AUTO, po zmianie na #SCALEMODE_LAYER
                                ;po próbie zmiany wielkości okna nastąpi zwiecha systemu.
    Sizeable=True,              ;Okno skalowalne, obszar za który można "złapać" jest podany poniżej,
                                ;gdybym tego nie zrobił, domyślnie jest ustawiony dolny prawy róg okna,
                                ;ale w tym wypadku jest niedostępny bo zegar ma kształt okręgu i nie da się
                                ;w gadżet skalowania kliknąć, gadżet taki o ile go w jakiś sposób nie narysujemy
                                ;będzie niewidoczny.
    SizeRegion = {{Type = #BOX, X=300, Y=300, Width = 100, Height = 100}}, ;obszar skalowania w dolnej prawej
                                ;ćwiartce tarczy zegara, zmienia swój
                                ;rozmiar wraz ze zmianą wielkości "okna".
    KeepProportions = True      ;zawartość okna zachowa proporcje, a przynajmniej powinna, bo niestety nie działa.
}

;DebugOutput(True)             ;zakomentowna funkcja do debugowania, przydaje się bardzo podczas szukania błędów.

```

"Tools:Hollywood/ARTKI/H3_Czas/H3_Czas.hws" Line: 34/195 Col: 1 Num LS Char: 10

Zrzut 2

SetFormStyle(#NORMAL) – przywracanie ustawienia surowego, czyli bez wygładzania, cienia i obwódki.

Możemy użyć kilku jednocześnie, ale musimy to zrobić za każdym razem przywołując funkcję SetFormStyle, to znaczy nie możemy napisać:

SetFormStyle(#ANTIALIAS, #EDGE).

Jak już określiliśmy jak mają wyglądać nasze wektory to zaczniemy je rysować. Żeby stworzyć jakikolwiek wektor musimy przydzielić mu najpierw ścieżkę, służy do tego funkcja:

StartPath(1)

Przyjmuje ona tylko jeden argument i jest to identyfikator jaki mu nadamy. Jak już mamy „pojemnik” na nasz wektor to możemy zacząć dodawać do niego krzywe. W naszym wypadku to będzie okrąg, stworzymy go za pomocą funkcji:

AddCircleToPath(1, z_CX, z_CY, 250)

Funkcja to wymaga czterech argumentów, pierwszy to IDentyfikator ścieżki w jakiej chcemy stworzyć wektora, w tym wypadku 1, drugi i trzeci to współrzędne środka okręgu, u nas to z_CX i z_CY czyli 250 i 250 pikseli. Czwarty argument to promień, w tym wypadku 250 pikseli.

Jak już skończyliśmy dodawać krzywe do ścieżki to musimy ją zamknąć, w innym wypadku jeśli zaczęlibyśmy dodawać nowe punkty zostałyby one

połączone z krzywą zdefiniowaną poprzednio, i mielibyśmy brzydką linię łączącą je ze sobą. Służy do tego funkcja:

ClosePath(ID)

Kilka słów na temat jak system ścieżek i krzywych (wektorów) działa.

Musimy pamiętać, że o ile nie użyjemy odpowiednich funkcji, to wszystko będzie rysowane od punktu 0,0 czyli od górnego lewego rogu. Dlatego też podaliśmy środek naszego okręgu jako z_CX i z_CY, w innym wypadku mielibyśmy widoczną jedynie dolną prawą ćwiartkę widoczną. Poza tym jeśli zdefiniujecie kilka krzywych w obrębie jednej ścieżki to wszystkie będą miały taki sam kolor, nie da się inaczej niż tworząc nową ścieżkę używając StartPath().

Ponad to rysując nowe krzywe w obrębie tej samej ścieżki będą one się zaczynały w punkcie gdzie skończyła się poprzednia, podając współrzędne nowego wektora będą one wskazywały na punkt ogólny naszego okna programu, a nie relatywnie do ostatniego punktu poprzedniej krzywej, chyba że użyjemy funkcji zaczynających się na **Rel**, są takie trzy: **RelCurveTo**, **RelLineTo** i **RelMoveTo**. Wtedy możemy podawać odległość względem ostatniego punktu.

Na przykład, ostatnia krzywa skończyła się w punkcie 100,100, teraz chcemy narysować linię która skończy się w 150, 100, użyjemy funkcji RelLineTo(1, 50, 0),

gdzie 1 to identyfikator ścieżki, a 50 i 0 to pięćdziesiąt pikseli na osi X, czyli w prawo i zero pikseli na osi Y, czyli ani w dół ani w górę. Gdybyśmy użyli funkcji `LineTo()`, to linia zostałaby narysowana nie od 100,100 do 150,100, tylko do 50,0.

Ok, jak już mamy naszego pierwszego wektora, to trzeba go jakoś narysować na ekranie naszej apki. Jako że w naszym wypadku ma to być tło to musimy najpierw przekształcić krzywą w pędzel używając funkcji:

PathToBrush(ID, {{ID, Color, X, Y}}

Jako pierwszy argument oczekuje ona podania identyfikatora pędzla do którego krzywa albo krzywe są przypisywane, stąd podwójne nawiasy klamrowe, Color to oczywiście kolor krzywej, w tym wypadku możemy podać inny kolor dla każdego wektora przypisywanego do pędzla. Co więcej styl wypełnienia i wygładzanie, cień itd. mogą być różne bo są określane w trakcie tworzenia danej krzywej. Nasza funkcja wygląda tak:

Obramowanie = PathToBrush(Nil, {{ID=1, Color=\$FFFFFF}})

No dobra, ale zapytacie o co kaman z tym Obramowaniem i Nilem? Jest to system automatycznego przypisywania identyfikatora, poprzez podanie stałej wbudowanej `Nil`; sama wartość identyfikatora jest określana przez Hollywood, my tylko podajemy nazwę pod którą ma być zapisana jego wartość, tu "Obramowanie".

Ten sposób może być używany wszędzie gdzie zajdzie taka potrzeba. Bezpośrednie podawanie wartości ID ogranicza nas do liczb, czasami ciężko się połapać co jest co, w takim wypadku przychodzi z pomocą automatyczne przypisywanie ID.

Dzięki temu możemy nadać identyfikatorowi jakąś znaczącą dla nas nazwę i używać jej podobnie jak zmiennych.

ID=1 to identyfikator ścieżki jaką przypisujemy do pędzla, i na koniec chcemy żeby nasz wektor miał kolor biały. Color=\$FFFFFF. Współrzędne X i Y pomijamy.

Dodam tylko jeszcze że mimo zamiany wektora na pędzel, wciąż pozostanie on wektorem, czyli obraca-

nie, czy zmiana rozmiaru nie będzie wpływać na jego jakość.

Cały ten zabieg z zamianą potrzebny nam był, żeby użyć wspomnianego wyżej triku, umożliwiającego stworzenie zegarka bez ramek okna i tła. Tło takie nie musi mieć żadnego regularnego kształtu, będzie wyglądać tak jak pędzel (brush) po zastosowaniu następnej funkcji, którą jest:

BrushToBGPic(BrushID, BGPicID)

Jako pierwszy argument podajemy identyfikator pędzla, u nas to będzie "Obramowanie", a jako drugi identyfikator tła, u nas "1". Następnie użyjemy funkcji do wyświetlania obrazka z tłem:

DisplayBGPic(ID {X, Y})

ID jak się domyślicie to identyfikator obrazka, a X i Y to współrzędne ekranu na którym nasz program będzie uruchamiany. Możemy użyć liczb albo stałych wbudowanych, takich jak `#TOP`, `#RIGHT`, `#CENTER` itp. Poza liczbami i wbudowanymi, możemy też użyć specjalnej stałej `#KEEPPOSITION`, przydaje się ona gdy zmieniamy tło i chcemy żeby nasza apka pozostała w tym samym miejscu na ekranie, bo w przeciwnym razie została by przesunięta na domyślną pozycję czyli środek.

Teraz otwieramy ekran apki stosując funkcję

OpenDisplay(1)

jedyny argument jaki przekazujemy do tej funkcji to identyfikator. Funkcja ta powoduje że ekran naszego programu staje się widoczny, Ta Da, i stała się magia. O innych argumentach które możemy przekazać do tej funkcji możecie poczytać w manualu, jest ich całkiem sporo. (Zrzut 3).

Mamy już tarczę zegara, teraz przyszedł czas na następne części naszego czasomierza.

Zajmiemy się punktami godzin i minut – romby i kreseczki. Samo tworzenie obiektów wektorowych, to pikuś, schody się zaczynają w miejscu gdy chcemy je automatycznie wyrysować tak żeby użyć tylko jednej pętli dla obydwu, i żeby jeszcze znalazły się na odpowiednich miejscach. Tu z pomocą przyszedł mi `teh_Kain`, z resztą nie tylko tu.

```

EscapeQuit(True)           ;Wyjście z aplikacji klawiszem ESCape
/***** ZMIENNE GLOBALNE *****/
z_Radius = 225             ;odległość punktów minut i godzin od środka zegara
z_CX = 250                 ;współrzędne środka zegara na osi X...
z_CY = 250                 ; i osi Y
/***** TARCZA ZEGARA *****/
Tarcza zegara jednocześnie będzie tłem naszej apki i wyznaczy początkowy rozmiar okna.
Dzięki takiemu rozwiązaniu mogłem pominąć atrybuty Width i Height w funkcji prapocessora @DISPALY.*

SetFillStyle(#FILLCOLOR)   ;ustawiam sposób wypełniania wektorów kolorem, może być z przezroczystością w formie ARGB
SetFormStyle(#ANTIALIAS)   ;włączam wygładzanie krawędzi wektorów

StartPath(1)               ;zaczynam ścieżkę ID=1
AddCircleToPath(1, z_CX, z_CY, 250) ;dodaję okrąg do ścieżki 1 o środku z_CX/z_CY i średnicy 250
ClosePath(1)               ;zamykam ścieżkę 1

Obramowanie = PathToBrush(Nil, {{ID=1, Color=$FFFFFF }})
BrushToBGPic(Obramowanie, 1)
DisplayBGPic(1, {X = #KEEPPOSITION, Y = #KEEPPOSITION})
OpenDisplay(1)

"Tools:Hollywood/ARTKI/H3_Czas/H3_Czas.hws" Line: 53/194 Col: 1 Num LS Char: 10

```

Zrzut 3

Należy pamiętać że wcześniej ustawione styl wypełnienia i styl formy #FILLCOLOR i #ANTIALIAS wciąż obowiązują, dopóki ich nie zmienimy. W naszym wypadku nie ma takiej potrzeby dlatego zostawiamy jak jest.

Zaczynamy od rozpoczęcia nowej ścieżki, identyfikator może być ten sam co poprzednio, bo ścieżka tarczy zegara jest już zamknięta. Identyfikator może a nawet musi mieć inną wartość jeśli mamy więcej niż jedną ścieżkę to wtedy musimy podać inny żeby program wiedział gdzie chcemy dodać nowe wierzchołki naszych wektorów. Ustawiamy punkt z którego zaczniemy kreślić naszą krzywą na 0,0, tak na wszelki wypadek. Służy do tego funkcja:

MoveTo(1, 0, 0)

Pierwszy argument to identyfikator ścieżki a dwa zera to współrzędne X i Y.

Teraz narysujemy pierwszą linię rombu, który będzie punktem godzin. Użyjemy:

LineTo(1, 8, 20)

Tak jak poprzednio, 1 to identyfikator ścieżki, a 8 i 20 to współrzędne X i Y do których prowadzi nasza linia, oczywiście swój początek ma na 0, 0 dzięki MoveTo().

Następnie użyjemy funkcji bardzo podobnej do LineTo(), jedyna różnica jest taka że nie rysuje linii do podanych współrzędnych, tylko względnie od ostatniego wierzchołka, czyli rysujemy linię od 8, 20

(ostatni wierzchołek poprzedniego wektora) do 0, 40 (który teraz jest punktem od którego zaczniemy rysować dalej)

RelLineTo(1, -8, 20)

Jak widać nazwa funkcji różni się jedynie przedrostkiem "Rel", od relative, przyjmuje takie same argumenty, ale zachowuje się inaczej niż LineTo(). Jest to wygodne bo możemy odnieść się względem ostatniego punktu, a nie wyliczać wszystko względem ogólnego układu współrzędnych naszego okna. Dwa polecenia RelLineTo więcej i „zamkniemy” nasz romb przez ClosePath(1).

Następnie przypisujemy nasze cztery proste tworzące romb do identyfikatora pędzla "Romb" jednocześnie nadając mu odpowiedni kolor, w tym wypadku szary.

Romb = PathToBrush(Nil, {{ID=1, Color=\$707070}})

Pamiętajcie o podwójnych nawiasach klamrowych, nawet jeśli przypisujecie brushowi jedną ścieżkę.

Teraz ze względu na udokumentowane aczkolwiek niechciane przeze mnie zachowanie Hollywooda musimy zmienić styl wypełnienia:

SetFillStyle(#FILLNONE, 1)

FillNone nie wypełnia powierzchni a jedynie krawędzie ścieżek, linią o podanej w drugim argumencie grubości, u nas 1. Dodam tylko że gdybyśmy nie

zmienili stylu wypełnienia, nasza pojedyncza linia nie zostanie narysowana jeśli SetFormStyle będzie ustawiony na #ANTIALIAS, bez niego linia będzie widoczna. Ma to miejsce jedynie w przypadku jednej linii, gdy są dwie lub więcej wszystko jest ok.

Szczerze pisząc nie wiem co autor Hollywooda miał na myśli tak to konstruując...

Dobra, dla mnie to bug, a jego czerstwe tłumaczenie nie ma dla mnie sensu :P.

Nie będę opisywał procesu kreślenia jednej linii, przejdę od razu do mięsa, czyli pętli liczącej współrzędne pozycji kreski i rombów i rysującej je w odpowiednich miejscach na tarczy zegara. Zadanie byłoby dużo łatwiejsze gdyby można przesunąć punkt uchwytu obiektu poza jego obręb. Niestety argumenty AnchorX i Y przyjmują jedynie wartości po między 0 i 1. Nic z tym nie mogłem zrobić przez co wszystko się skomplikowało. Nie ma tego złego i dzięki temu nauczyłem się czegoś nowego.

Przechodząc do rzeczy. Użyłem pętli **While - Wend** bo jej konstrukcja najlepiej mi odpowiadała (W żadnym wypadku nie twierdzę, że skrypt jest zoptymalizowany, jeżeli ktoś ma uwagi czy rady jak ulepszyć algorytmy zawarte w moim kodzie to dajcie znać). Ponad to wewnątrz pętli While musiałem użyć instrukcji warunkowej **If - ElseIf - Else - EndIf**. Pętla oblicza wszystkie punkty godzi i minut a instrukcja warunkowa IF decyduje czy ma zostać narysowany Romb godzin czy Kreska minut.

Zacznijmy lekcję matmy. Jako że mamy sześćdziesiąt minut w godzinie i tyle jest na tarczy, to cały okrąg podzieliłem przez 60, jak wiadomo pełny okrąg ma 360 stopni, więc dzieląc go przez 60 wychodzi nam jeden symbol Kreski albo Romba co 6 stopni. Teraz mając jedynie współrzędne środka okręgu $z_CX=250$ na osi X i $z_CY=250$ na osi Y i długość promienia okręgu na którym mają być rysowane punkty $z_Radius=225$, użyłem trygonometrii żeby obliczyć gdzie każdy punkt ma się znajdować.

Wygląda to tak:

```
i=0
While i < 60
    i=i+1
    z_angle = i*2*#Pi/60
```

$i=0$, zeruje wartość zmiennej i. **While i < 60** to początek pętli gdzie sprawdzamy czy zmienna "i" nie jest większa od 60, jeśli jest mniejsza to dodajemy do niej 1 - $i=i+1$. Następnie zmiennej **z_angle** przypisujemy wynik działania $i*2*#Pi/60$. W pierwszym przebiegu pętli wynik będzie następujący:

$1*2Pi/60 \Rightarrow 1*360/60 \Rightarrow 6$, w drugim $2*360/60 \Rightarrow 12$, a w ostatnim $60*360/60 \Rightarrow 360$.

Tu kilka uwag. Po pierwsze #Pi to stała Hollywood i oczywiście symbolizuje liczbę Pi.

Po drugie zwróćcie uwagę czy dzielenie jest przy użyciu znaku **"/** (**forward slash**), jest bowiem różnica po między **/** i ****. Wynikiem dzielenia przez **/** jest liczba zmiennoprzecinkowa, natomiast przy dzieleniu przez **"\"** (**backslash**) dostajemy jedynie wartość całkowitą z dzielenia, miejsca po przecinku są obcinane.

$$X = z_CX + z_Radius * \cos(z_angle)$$

$$Y = z_CY + z_Radius * \sin(z_angle)$$

Policzywszy kąt i mając daną długość promienia, możemy użyć funkcji trygonometrycznych do obliczenia współrzędnych punktu na krawędzi okręgu. Użyjemy do tego funkcji **Cos(z_angle)** i **Sin(z_angle)**, wynikiem jest wartość w Radianach, którą trzeba zamienić na stopnie funkcją **Deg()** jednocześnie przypisując wynik zmiennej **z_rotation**

$$z_rotation = \text{Deg}(z_angle)$$

Nie będę się zagłębiał w matematyczny aspekt bo to musicie sami rozgryźć, a poza tym istnieje duże ryzyko, że coś pokręcę i będzie wstyd.

Założenie jest takie:

Jeśli kąt w danym przebiegu pętli różny jest od 30 stopni to rysuj Kreskę

Jeśli nie to rysuj Romb

Prościej być nie może. Musimy to jedynie zapisać używając funkcji Hollywood.

Wartość 30 bierze nam się z dzielenia 360 (pełen okrąg) przez 12 (ilość godzin).

Ale jak określić czy wartość kąta to 30, 60 czy 150 stopni, nie pisząc dla każdej wartości po między 60 i 360 stopni instrukcji warunkowej ElseIf? Możemy to obliczyć sprawdzając czy reszta z dzielenia przez 30 da nam 0. Bo $60/30=2$ czyli nie ma reszty.

W tym celu użyjemy operatora %, który jako wynik podaje wartość reszty z dzielenia. Na przykład $7\%3$ da nam 1, $5\%3$ da nam 2.

Niestety nie miałem tak łatwo bo okazało się, że reszta z dzielenia w punktach gdzie mają być godziny częściej równa jest 29 niż 0. Udało mi się pokonać i tą trudność stosując operator logiczny **AND**.

Operator $\langle \rangle$ znaczy „różny od”. O tych operatorach jak i wielu innych poczytacie w manualu.

Nasz blok warunkowy wygląda tak:

```
If (z_rotation%30)<>0 AND (z_rotation%30)<>29 ;
jeśli reszta z dzielenia różna (<>0) od 0 i (<>29) od
29
```

```
    DisplayBrush(Kreska, X, Y, {Rotate=270-z_
rotation, AnchorX=0.5, AnchorY=1, Shadow=True,
ShadowColor=$707070, ShadowSize=0})
```

```
Else
```

```
    DisplayBrush(Romb , X, Y, {Rotate=270-z_ro-
tation, AnchorX=0.45, AnchorY=0.5, Shadow=True,
ShadowColor=$707070, ShadowSize=0})
```

```
EndIf
```

Wend - oznaczenie końca pętli While - Wend.

Teraz wyjaśnię co się dzieje w Funkcji rysującej **DisplayBrush()**. Pierwszy argument to identyfikator pędzla (brusha) do wyświetlenia, dwa następne to współrzędne na osiach X i Y. Teraz mamy {Tabelę} z listą argumentów. Pełna lista wiecie gdzie jest, tak, w manualu ;p. Pierwszy element tabeli **Rotate=270-z_rotation** to wartość w stopniach o jaką ma być obrócony pędzel wokół uchwytu (Anchor), wartość ta liczona jest w pętli trochę wyżej. **270** to stopnie o jakie musiałem obrócić każdy z pędzli bo były pod złym kątem. Pozycje **AnchorX i Y** zmieniają punkt uchwytu i obrotu obiektu.

Shadow=True to włączenie cienia, ShadowColor chyba nie muszą tłumaczyć, ale już ShadowSize muszą, bo to nie jak wskazuje nazwa rozmiar cienia a jego odległość od rzucającego go obiektu, w tym wypadku pędzla. Ja ustawiłem na 0 bo nie chodziło mi o cień, a o efekt poświaty wokół pędzla (*Zrzut 4*).

Teraz tworzymy wskazówki, postępujemy podobnie jak w wypadku Kresek i Rombów, nie będę tego

```

/***** PUNKTY MINUT i GODZIN *****/

StartPath(1)
MoveTo(1, 0, 0)          ;ustawiamy punkt z którego zaczniemy kreślić wektor na 0,0
LineTo(1, 8, 20)          ; \      8 pikseli w lewo i 20 w dół
RelLineTo(1, -8, 20)       ; /      8 pikseli w prawo i 20 w dół
RelLineTo(1, -8, -20)      ; \      8 pikseli w prawo i 20 w górę
RelLineTo(1, 8, -20)       ; /      8 pikseli w lewo i 20 w górę
ClosePath(1)
Romb = PathToBrush(1, {{ID=1, Color=$707070}}) ;zapisujemy naszą ścieżkę jako pędzel "Romb"

SetFillStyle($FILLNONE,1) ;tu musimy zmienić styl wypełnienia na samą krawędź bo Hollywood
                           ;nie potrafi rysować wektora składającego się z pojedynczej linii,
                           ;gdzy SetFormStyle jest ustawiony na Antialias. Nie wiem dlaczego,
                           ;twórca Hollywood napisał że to jest udokumentowane zachowanie i
                           ;tak ma być kropka...

StartPath(1)
MoveTo(1, 0, 0)
LineTo(1, 0, 20)
ClosePath(1)
Kreska = PathToBrush(1, {{ID=1, Color=$707070}})

i=0                      ; nadajemy zmiennej i wartość zero
While i < 60              ;gdzy "i" mniejsze od 60, czyli może przyjąć wartości od 0 do 59

    i=i+1                  ;to "i" równa się "i" plus jeden, zwiększamy wartość "i" przy każdym
                           ;wykonaniu pętli, do póki warunek i<60 jest prawdziwy.

    z_angle = i*(2*#Pi/60)
    X = z_CX + z_Radius * Cos(z_angle)
    Y = z_CY + z_Radius * Sin(z_angle)

    z_rotation = Deg(z_angle)

    If (z_rotation%30)<>0 AND (z_rotation%30)<>29
        DisplayBrush(Kreska, X, Y, {Rotate=270-z_rotation, AnchorX=0.5, AnchorY=1, Shadow=True, ShadowColor=$707070, ShadowSize
    Else
        DisplayBrush(Romb , X, Y, {Rotate=270-z_rotation, AnchorX=0.45, AnchorY=0.5, Shadow=True, ShadowColor=$707070, ShadowSi
    EndIf

Wend

"Tools:Hollywood/ARTKI/H3_Czas/H3_Czas.hws" Line: 104/194 Col: 52 Num LS Char: 10

```

Zrzut 4

opisywał, bo nie ma tam nic nowego. Jedyne co wymaga wyjaśnienia to funkcja **FreePath(1)**. Powoduje ona usunięcie z pamięci nie potrzebnej nam już ścieżki.

Teraz nadszedł czas na zajęcie się czasem, a jaśniej pisząc wyciągniemy sobie wartość godzin, minut i sekund z systemowego zegara funkcjami **GetTime()** i **MidStr()** i przypiszemy je odpowiednim zmiennym.

```
z_czas = GetTime(True)
```

z_czas = GetTime(True) z_czas to zmienna która przechowa systemowy czas, podanie True spowoduje, że czas będzie zawierał także sekundy, w innym wypadku są to jedynie godziny i minuty. Format w jakim przekazywana jest godzina wygląda tak: **HH:MM:SS**.

Teraz żeby zmiennym z_godz, z_min i z_sek przypisać wartości odpowiednio HH, MM i SS musimy użyć funkcji operującej na ciągach znaków MidStr(). Jako jej pierwszy argument przekazujemy zmienną pod którą jest nasz ciąg znaków, u nas to jest z_czas gdzie przechowujemy HH:MM:SS, drugi argument to pozycja znaku w ciągu od którego chcemy zacząć kopiowanie znaków, jeśli jest to pierwszy znak to podajemy 0, a trzeci argument funkcji to ilość znaków jakie nas interesują. W przypadku godziny mamy:

z_godz = MidStr(z_czas, 0, 2) - ze zmiennej z_czas skopiuuj dwa znaki zaczynając od pierwszego i wklej je do zmiennej z_godz.

```
z_godz = MidStr(z_czas, 0, 2)
z_min = MidStr(z_czas, 3, 2)
z_sek = MidStr(z_czas, 6, 2)
```

Teraz musimy te wartości zamienić na kąty pod jakimi mają być rysowane wskazówki naszego zegara.

W przypadku sekund sprawa jest prosta, mnożymy wartość zmiennej z_sek razy sześć, to sześć to kąt co ile są rysowane Kreski minut na tarczy zegara. Musimy jeszcze tylko pomnożyć tą wartość razy minus jeden bo w przeciwnym wypadku wskazówki będą nam chodziły „pod prąd”.

```
z_sek_angle = (z_sek*6)*-1
```

W przypadku minut postępujemy podobnie, z tym że musimy podać kąt pośredni po między sąsiadującymi minutami, w innym wypadku wskazówka się nie

poruszy dopóki nie nastąpi zmiana minuty na następną, zwiększamy jakby rozdzielczość ruchu wskazówki. W wypadku wskazówki sekund ruch jest skokowy, bo funkcja GetTime() nie umożliwia odczytu milisekund z systemu. Można by samemu to policzyć, aczkolwiek czas i tak byłby niedokładny w stosunku do systemowego, bo byłby zaokrąglany do sekundy w której uruchamiamy nasz program. Zwróćcie uwagę, że wartość zmiennej zwiększającej „rozdzielczość” ruchu minutowego ramienia, to jest obliczona jedną linie wyżej wartość kąta sekundnika, a nie sekundy odczytane funkcją GetTime().

```
z_min_angle = ((z_min*6)*-1)+ z_sek_angle/60
```

Z godzinami jest bardzo podobnie, z tym że jest ich pięć razy mniej niż minut na tarczy więc mnożymy godzinę przez trzydzieści zamiast sześciu. Poza tym także w tym wypadku używamy wyliczoną wartość kąta wskazówki minutowej i dzielimy ją przez ilość godzin na tarczy zegara, czyli dwanaście.

```
z_godz_angle = ((z_godz*30)*-1)+ z_min_angle/12
```

Teraz już tylko pozostało wyświetlić to co udało nam się policzyć. Używamy do tego tej samej funkcji co w przypadku innych elementów zegara, czyli DisplayBrush.

Składnia jest identyczna, oczywiście poza identyfikatorem pędzli i kąta obrotu (*Zrzut 5*).

Jak pewnie zauważyliście nasz zegar „nie chodzi”. Żeby tchnąć w niego życie napiszemy sobie funkcję prywatną, zwaną też funkcją użytkownika. Według zaleceń jej nazwa powinna rozpoczynać się przedrostkiem p_ żeby odróżnić ją od funkcji wbudowanych i uniknąć konfliktu w razie gdyby funkcja o takiej nazwie została dodana przez autora Hollywood w przyszłości.

Żeby poinformować Hollywood, że chcemy napisać własną funkcję, definicję jej zamykamy po między **Function** i **EndFunction**

A oto jak będzie wyglądać nasza funkcja:

Pierwszy krok to usunięcie wskazówek zegara, które są wyświetlone za sprawą trzech rozkazów DisplayBrush znajdujących się poza naszą funkcją. Funkcja


```

/***** WSKAZÓWKA GODZIN *****/
SetFillStyle(#FILLCOLOR)
StartPath(1)
MoveTo(1, 0, 0)
LineTo(1, 10, 40)
RelLineTo(1, -10, 100)
RelLineTo(1, -10, -100)
RelLineTo(1, 10, -40)
ClosePath(1)
WskazMala = PathToBrush(Hil, {{ID=1, Color=$707070}})

/***** WSKAZÓWKA MINUT *****/
StartPath(1)
MoveTo(1, 0, 0)
LineTo(1, 10, 40)
RelLineTo(1, -10, 160)
RelLineTo(1, -10, -160)
RelLineTo(1, 10, -40)
ClosePath(1)
WskazDuza = PathToBrush(Hil, {{ID=1, COLOR=$707070}})

/***** WSKAZÓWKA SEKUND *****/
SetFillStyle(#FILLNONE, 2) ;dwójka określa grubość wskazówki
StartPath(1)
MoveTo(1, 0, 0)
LineTo(1, 0, 200)
ClosePath(1)
WskazSek = PathToBrush(Hil, {{ID=1, COLOR=$6088C8}})
FreePath(1)

/***** RYSUJEMY WSKAZÓWKI WEDŁUG CZASU *****/
z_czas = GetTime(True) ;zapisuję w zmiennej z_czas aktualną godzinę wraz z sekundami w formie HH:MM:SS
z_godz = MidStr(z_czas, 0, 2) ;zapisuję w zmiennej z_godz wartość HH z pobranego czasu
z_min = MidStr(z_czas, 3, 2) ;zapisuję w zmiennej z_min wartość MM z pobranego czasu
z_sek = MidStr(z_czas, 6, 2) ;zapisuję w zmiennej z_sek wartość SS z pobranego czasu

z_sek_angle = (z_sek*6)*-1
z_min_angle = ((z_min*6)*-1)+ z_sek_angle/60
z_godz_angle = ((z_godz*30)*-1)+ z_min_angle/12

DisplayBrush(WskazMala, z_CX, z_CY, {Rotate=z_godz_angle, AnchorX=0.5, AnchorY=1, Shadow=True, ShadowColor = $00707070, Sha
DisplayBrush(WskazDuza, z_CX, z_CY, {Rotate=z_min_angle, AnchorX=0.5, AnchorY=1, Shadow=True, ShadowColor = $00707070, Shad
DisplayBrush(WskazSek, z_CX, z_CY, {Rotate=z_sek_angle, AnchorX=0.5, AnchorY=1, Shadow=True, ShadowColor = $6088C8, ShadowS

```

Zrzut 5

nasza będzie przyzywana raz po upływie każdej sekundy, przez co gdyby wskazówki nie zostały narysowane przed wywołaniem naszej prywatnej funkcji, to po uruchomieniu naszego programu zegar nie miałby wskazówek przez jedną sekundę.

```

Undo(#Brush, WskazMala)
Undo(#Brush, WskazDuza)
Undo(#Brush, WskazSek)

```

Funkcja **Undo()** usuwa obiekt graficzny z ekranu, jako pierwszy argument przekazujemy typ obiektu do usunięcia, używamy do tego **specjalnych stałych**, w naszym wypadku jest to **#BRUSH**. Drugi argument to identyfikator obiektu. Trzeci to poziom zagnieżdżenia.

Dla przykładu gdy mamy wyświetlone trzy te same obiekty i chcemy usunąć tylko ostatni, to podajemy tu wartość 1 (domyślna), każdy poprzedni obiekt ma wartość o jeden większą, czyli gdybyśmy chcieli

usunąć pierwszy narysowany z trzech widocznych obiektów to podajemy wartość 3.

Drugi krok, to ponowne odczytanie czasu systemowego i przeliczenie go na odpowiednie kąty dla naszych wskazówek.

```

z_czas = GetTime(True)
z_godz = MidStr(z_czas, 0, 2)
z_min = MidStr(z_czas, 3, 2)
z_sek = MidStr(z_czas, 6, 2)

```

```

z_sek_angle = (z_sek*6)*-1
z_min_angle = ((z_min*6)*-1)+ z_sek_angle/60
z_godz_angle = ((z_godz*30)*-1)+ z_min_angle/12

```

Trzeci i ostatni krok to wyświetlenie ich. I tak w kółko.

Może dziwić takie rozwiązanie ale takie zastosowałem bo lepszego w tej chwili nie znam.

Nasza funkcja prywatna będzie wywoływana w odstępach określonych w funkcji **SetInterval()**, która natomiast będzie wywoływana przez funkcję **WaitEvent()** która to znajduje się w nieskończonej pętli **Repeat - Forever**.

SetInterval(1, p_RuchWskazowek, 1000)

Przekazujemy do tej funkcji trzy argumenty, pierwszy to identyfikator interwału, czyli odstęp czasu, drugi to funkcja jaka będzie wywoływana co podaną jako trzeci argument ilość milisekund.

SetInterval() nie będzie działać bez użycia WaitEvent(), która jest bardzo ważną funkcją w Hollywood, bez niej funkcja przekazana w SetInterval nie będzie wywoływana.

WaitEvent() ma tą zaletę, że jest wykonywana co jakiś czas, czyli raz na ilość milisekund podanych w funkcji SetInterval(). Dzięki temu możemy ograniczyć szybkość wykonywania się głównej pętli naszego programu do pożądanej ilości klatek na sekundę bez względu na szybkość maszyny na której nasza apka jest uruchamiana. Bez niej program wykonywany byłby z maksymalną możliwą szybkością, pożerając tym samym sto procent czasu procesora.

Żeby funkcja WaitEvent() była wywoływana tak długo jak działa nasz program, umieszczamy ją w nieskończonej pętli:

Repeat WaitEvent() Forever

W naszym wypadku nie przekazujemy żadnych argumentów do tej funkcji. WaitEvent sama się zatroszczy o wywołanie funkcji podanej jako argument SetInterval (Zrzut 6).

I tym sposobem dotarliśmy do końca trzeciego odcinka bojów z Ostrokrzewem, zegar działa choć sporo mu brakuje, resztą rzeczy zajmę się w następnych odcinkach, bo to jeszcze nie koniec, a wręcz dopiero początek.

Nie wiem na ile możliwości Hollywood i moje zdolności do jego nauki mi pozwolą osiągnąć, bo musicie pamiętać, że ten cykl to swoisty wykres prezentujący krzywą pozyskiwania wiedzy i doświadczenia w posługiwaniu się nim. Zdaję sobie sprawę, że jak do tej pory szalu nie ma. Zegarek i to w dodatku kopia systemowego raczej jest mało porównawczy, ale od czegoś trzeba zacząć. Na wodotryski przyjdzie czas, miejmy nadzieję ;).

Pliki ze skryptem i wykonywalne plus aplet, są do ściągnięcia z tego samego miejsca co poprzednio:

www.czernik.me/hollywood.html

```
Function p_RuchWskazowek ()
    Undo(#Brush, WskazMala)
    Undo(#Brush, WskazDuza)
    Undo(#Brush, WskazSek)

    z_czas = GetTime(True)           ;zapisuję w zmiennej czas$ aktualną godzinę wraz z sekundami w formie HH:MM:SS
    z_godz = MidStr(z_czas, 0, 2)    ;zapisuję w zmiennej godz$ wartość HH z pobranego czasu
    z_min = MidStr(z_czas, 3, 2)     ;zapisuję w zmiennej min$ wartość MM z pobranego czasu
    z_sek = MidStr(z_czas, 6, 2)     ;zapisuję w zmiennej sek$ wartość SS z pobranego czasu

    z_sek_angle = (z_sek*6)*-1
    z_min_angle = ((z_min*6)*-1)+ z_sek_angle/60
    z_godz_angle = ((z_godz*30)*-1)+ z_min_angle/12

    DisplayBrush(WskazMala, z_CX, z_CY, (Rotate=z_godz_angle, AnchorX=0.5, AnchorY=1, Shadow=True, ShadowColor = $00707070, Sha
    DisplayBrush(WskazDuza, z_CX, z_CY, (Rotate=z_min_angle, AnchorX=0.5, AnchorY=1, Shadow=True, ShadowColor = $00707070, Shad
    DisplayBrush(WskazSek, z_CX, z_CY, (Rotate=z_sek_angle, AnchorX=0.5, AnchorY=1, Shadow=True, ShadowColor = $6088C8, ShadowS

EndFunction

SetInterval(1, p_RuchWskazowek, 1000)

Repeat
    WaitEvent()
Forever

End()
```

"Tools:Hollywood/ARTKI/H3_Czas/H3_Czas.hws" Line: 186/190 Col: 7 Num LS Char: 10

Zrzut 6

Jabłko pod prądem

czyli naprawiamy zasilacz do iBooka/PowerBooka G4

recedent

Załóżmy, że jesteś wrednym mosi-
stą, który korzysta na co dzień ze
szrotowego sprzętu od znanej
firmy, sprzedającej zbyt drogie,
rebrandowane pecety. Na domiar
złego, Twój sprzęt został sprze-
dany z horrendalną marżą już
ponad dekadę temu (dla nieogarów
– chodzi o iBooka G4 lub Power-
Booka G4). Twoją codzienną sie-
lankę przerywa jednak pewnego
dnia awaria zasilacza, a konkretnie
– jego kabla, który urwał się sam,
przegryzł chomik siostrzeńca, albo
odcięła nożem zazdrosna o Twoją
uwagę dziewczyna. Pół biedy, jeśli
rozwałił się pośrodku (kable
można wyizolować, skrócić, zluto-
wać, i założyć koszulkę), ale z
reguły usterka ma miejsce w
dwóch kluczowych miejscach –
albo przy samym zasilaczu, albo
przy samej końcówce, a wtedy bez
konkretnej dłubaniny się nie obej-
dzie.

Druga przykrość, która mogła Cię
spotkać, to kompletny pad zasila-
cza, lub zakup komputera bez
niego. Kupno nowego może kon-
kretnie nadszarpnąć nasz budżet,
gdyż użytkownicy potrafią w ekstremal-
nych przypadkach kosztować nas
równowartość banknotu z Włady-
sławem Jagiełłą. Można oczywiście
kupić chiński zamiennik, ale
z doświadczenia wiem, że potrafi
zamiast obiecanych 65 W dawać
ich około 24. Ostatecznym rozwi-
ązaniem może się wtedy okazać

zakup sprzętu z kategorii
„sprawny, ale zmalretrowany”, z
którego przy niewielkim skillu i
wskazówkach z niniejszego tuto-
riała zrobimy niemal „funkiel
nówkę, Jobs płakał jak sprzedawał,
do dzisiaj dzwoni posłuchać jak
pyrczy pod obciążeniem”. Zatem
– do dzieła!

**Wariant 1 - kabel urwany przy
obudowie zasilacza**, a następnie
zgubiony, spalony, zjedzony i
wydalony w nieznanym nam miej-
scu, ewentualnie urwana i zagu-
biona w akcji końcówka:



Opisywany wariant jest prostszy,
dlatego idzie na rozgrzewkę.
Odpowiedni kabel wystarczy:
przełożyć z drugiego, niespraw-
nego zasilacza, lub nabyć na
znanym portalu aukcyjnym za kil-
kanaście złotych. W pierwszym
przypadku zachowamy fikuśny

ficzer „świecącej obrączki”, infor-
mującej czy komputer się ładuje,
czy już nie. W drugim – niestety
mamy do czynienia z chińskim
zamiennikiem, który co prawda
działa bez zarzutu, ale świecić już
nie świeci (potrzebna jest do tego
dodatkowa elektronika i diody,
których nasi serdeczni przyjaciele
z Państwa Środka nie zdecydowali
się zmieścić w kuszącej swym
krągłym kształtem wtyczce).



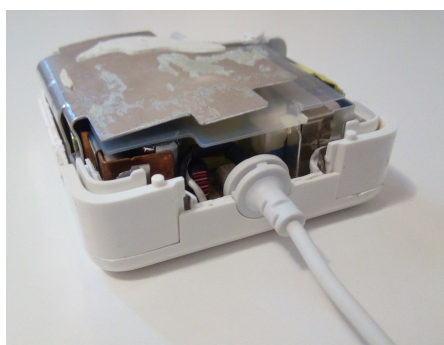
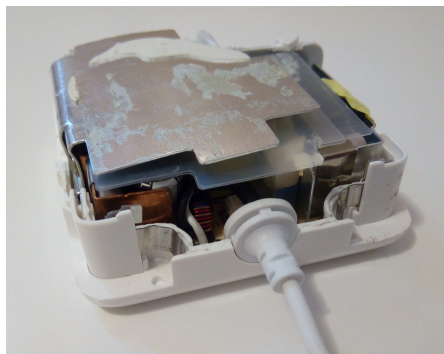
Zanim jednak weźmiemy się za
lutowanie kabla, należy rozwalić
zasilacz. Potrzebne nam będą do
tego dwie sprawne ręce, szczypce
uniwersalne i trochę brutalnej,
zwierzęcej siły. Najskuteczniejsza
metoda otwarcia naszej „puszki”
to najpierw odchylić zaczep do
zwijania kabla, a następnie włożyć
w powstałą wnękę końcówkę
szczypiec i rozchylić je.



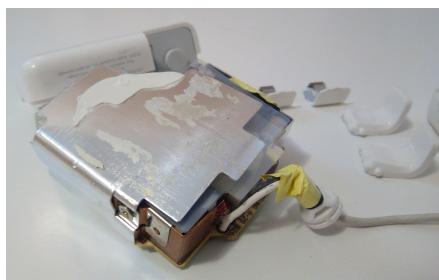
W tym właśnie momencie przyda się trochę siły, ale przyłożonej z umiarem – pudło jest solidnie sklejone w środku i może się zdarzyć, że „spęka”. Operację powtarzamy dla drugiej strony, potem delikatnie podważamy jakimś płaskim śrubokrętem i już powinniśmy mieć dostęp do wnętrza zasilacza. Uwaga, zaczepy do zwijania kabla oczywiście wylecą podczas tej operacji, lepiej pozbierać wszystkie plastiki i blaszki, żeby potem się nie zastanawiać gdzie przepadły.



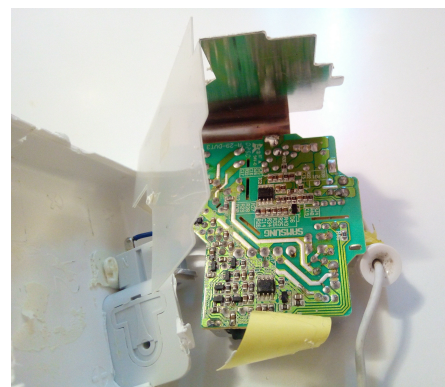
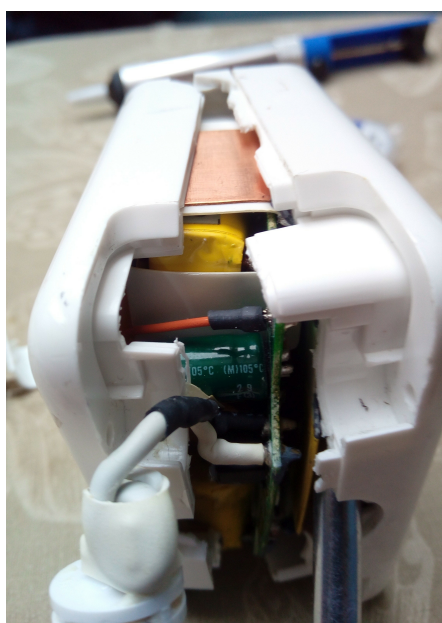
Dobrze też zapamiętać co, gdzie i w jaki sposób było włożone.



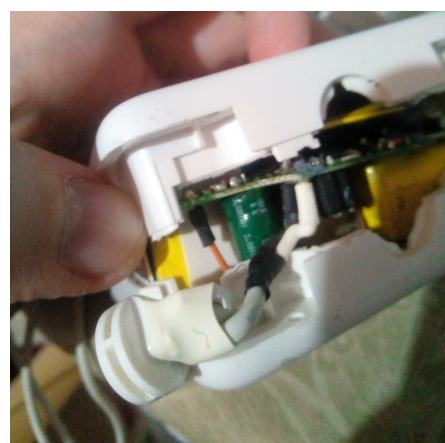
W środku znajdziemy oczywiście bebechy, zawinięte dodatkowo w aluminiową blachę izolującą. W niektórych rewizjach jest ona zalutowana, w innych – sklejona. Raczej nie obejdzie się bez jej odginania, choć jeśli jesteście bardzo zdolni...



Skoro widzimy już gdzie kończy się kikut naszego kabla, pora na małe rozlutowanko. Rozgrzewamy cynę w miejscach lutowania i wyciągamy niepotrzebną końcówkę szczypcami. Drobną uwagę – warto zapamiętać w którym miejscu znajdował się kabel czarny, a w którym szary. Szkoda laptopa, który prawdopodobnie by „puffnął” gdybyśmy podłączyli nowy kabelek odwrotnie.



Pora na przylutowanie naszego nowego kabla w oczyszczone w poprzednim kroku miejsca i złożenie wszystkiego z powrotem do kupy. Stary klej oczywiście nie da rady ponownie związać obudowy w jedną część, więc wypada użyć drobnego wspomagania. Poradniki na sieci polecają cyjanoakrylat, czyli popularną „Kropelkę”, ja jednak postanowiłem użyć dwuskładnikowego, przezroczystego kleju epoksydowego. Po złożeniu całości założyłem ścisk stolarski i nałożyłem warstwę kleju wzdłuż „szwów” zasilacza (omijając części ruchome, czyli – w naszym przypadku – zaczepy do zwijania kabla).



Zaraz zaraz, czy nie zapomniałem o czymś? Ach tak, wypadłoby może przetestować czy nasza przeróbka działa! Z serca doradzam zrobić to zanim zaczniemy szaleć z klejem.

Wariant 2 - kabel urwany przy samej wtyczce:



Pierwsze co musimy zrobić w takim przypadku to uwolnić końcówkę wtyczki, czyli zdjąć z niej białą, plastikową osłonkę. Ja użyłem do tego celu przecinaka, ale słyszałem też o metodzie „bez

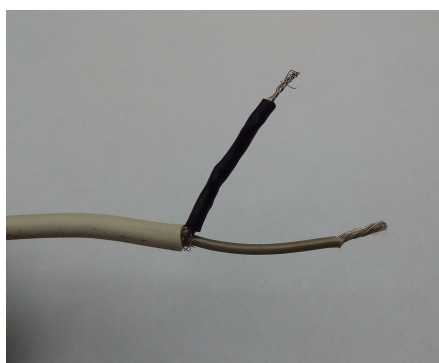


cięcia”, z wykorzystaniem wąskiego ostrza (żeby oddzielić świecący pierścień od osłonki) i zębów (żeby przytrzymać końcówkę w ustach, podczas gdy będziemy zdejmowali osłonę palcami). W każdym wypadku, po ściągnięciu osłony naszym oczom ukaże widok jak na obrazku.

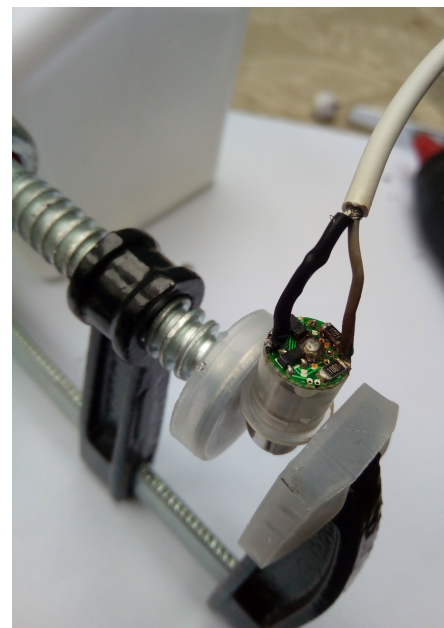


Jak widać, wewnątrz wtyczki mamy płytkę z dolutowanymi z dwóch stron przewodami z naszego kabelka. Zgodnie z przewidywaniami, jeden z nich będzie szary, a drugi czarny. Warto już teraz zaznaczyć sobie/zrobić zdjęcie który gdzie idzie, żeby później nie musieć rozbebeszać kolejnej końcówki dla porównania. Wygodę odlutowywania wydatnie zwiększy delikatne zsuniecie przezroczystego, plastikowego pierścienia w stronę końcówki. Teraz już można uwolnić pady od niepotrzebnych, urwanych kabli.

Nadeszła pora na przygotowanie drugiej części urwanego kabla (zakładając, że nie został zgubiony, spalony, zjedzony i wydalony w nieznanym nam miejscu). Po ściągnięciu kilku centymetrów izolacji okazuje się, że wewnątrz znajdziemy przewód szary, miedziany opłot i jakieś nylonowe włókna, których oczywiście wypada się pozbyć. Opłot skręcamy, a następnie elegancko zaizolowujemy, choćby rurką termokurczliwą.



Końcóweczki posrebrzamy, a następnie dolutowujemy do odsłoniętych wcześniej padów (jeśli osłonę zdjęliśmy bez jej rozcinań, dobrym pomysłem będzie założenie jej na kabel zanim zaczniemy lutować)



Reszta, czyli zabezpieczenie wtyczki to już kosmetyka. Używamy epoksy, kleju na gorąco, srebrnej taśmy – co kto woli. Oczywiście wypada wykonać uprzednio test poprawności działania.



Jak widać na załączonym obrazku – wszystko się pięknie ładuje, można zatem usiąść i napawać się dobrą robotą, sącząc powoli przez słomkę sok porzeczkowy (przy tym można na przykład zajrzeć na PPA).

To już wszystko w bieżącym odcinku, do zobaczenia w przyszłości!

Szaty motyla

Korni

Temat wyglądu blatu ekscytował amigowców od zawsze. Zaczęło się nieśmiało od standardowo konfigurowalnych 4 kolorów palety systemu.

Użytkownicy uznali, że to za mało. Szybko pojawiły się programy do ustawienia obrazka na „tapetę” czy wyboru czcionki.

Później, znowu dzięki użytkownikom powstały opcje zmiany wyglądu okna, nowe standardy ikon, nowe GUI w postaci Magic User Interface.

Obecnie MorphOS oferuje duże możliwości kreacji wizerunku.

Wygląd MorphOSa można podzielić na trzy części:

1. Skin

Systemowe *MOSSYS:Prefs/Skins/* oraz użytkownika *SYS:Prefs/Skins/*

Stworzenie katalogu *SYS:Prefs/Skins/MojaSkora* spowoduje, że w preferencjach systemu pojawi się nowa skórka o nazwie *MojaSkora*. W kompletnej skórcie znajdują się tam pliki konfiguracyjne oraz graficzne, bez nich zobaczymy podstawowy wygląd okien.

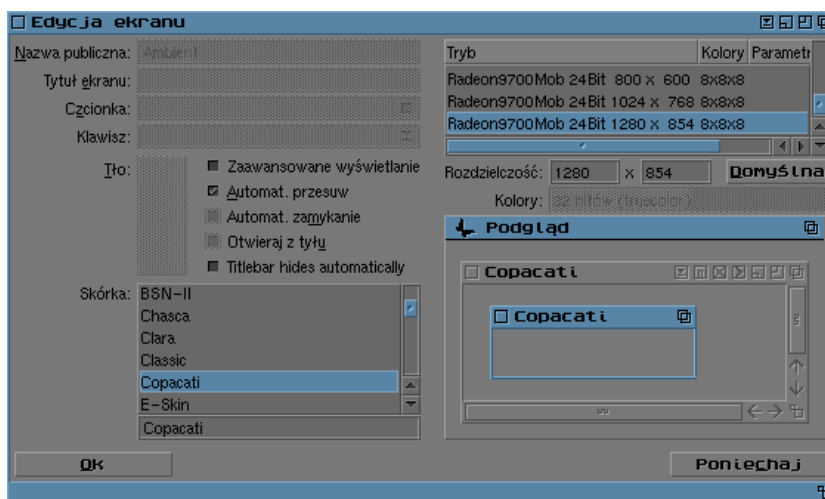
SkinConfig to tekstowy plik konfiguracyjny, nie tworzy się automatycznie, więc trzeba pożyczyć już gotowy z dowolnej systemowej skórki.

Ustawia się tam – kolory: systemowe, użytkownika, okna, menu; efekty menu; krawędzie okien; rozjaśnianie/ściemnianie krawędzi okien; rozmiary skórek; wygląd suwaków.

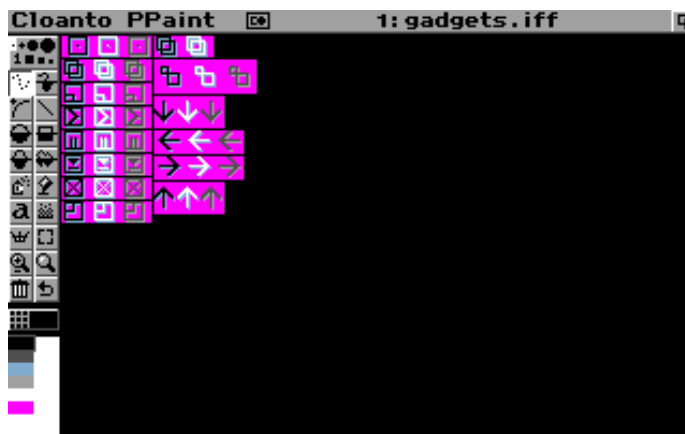
Efekty przezroczystości działają najlepiej przy korzystaniu z „zaawansowanego wyświetlania” (*Zrzut 1*).

Obrazki mogą być namalowane dowolnym programem obsługującym zapis do formatu obsługiwane przed datotypy MorphOS-a.

Kolor przezroczysty bądź kanał alpha może być użyty np. w PNG lub TIFF. W przypadku IFF ILBM kolor tła, który ma być przezroczysty, musi być różowy (R:255 G:0 B:255). (*Zrzut 2*)



Zrzut 1 - Okno wyboru skórek



Zrzut 2 - Malowanie gadżetów w programie PPaint

MojaSkora/ MenuTitleBar – tło tytułu menu belki ekranowej.

Prop#? – obrazki suwaka w 3. stanach.

STitleBarBackgroud – tło belki ekranowej.

STitlebarObject – znaczek belki ekranowej.

SysIclass#? – gadżety okien oraz znaczki w menu.

Obrazki zawierają maksymalnie 3 stany: aktywny, kliknięty, nieaktywny.

WindowTitlebar#? – tła okna.

WindowTitlebar#?Active – aktywne tła okna.

WindowShadow#? – cienie pod oknami; nie wyświetlają się z tradycyjnym wyświetlaniem.

Requesters/ – ikonki komunikatów.

Crash

DiskBusy

DiskError

DiskInsert

DiskNotValidated

DiskProtected

NoDisk

NotADOSDisk

Obrazki dla różnych wielkości powinny być umieszczone w podkatalogach np. *SYS:Prefs/Skins/MojaSkora/16/ 18/* itd.

Rozmiar ten dyktowany jest przez opcję *SKINHEIGHT*, w przypadku jej braku obrazki będą wczytywane z katalogu głównego skórki, a użyta czcionka ekranowa może powiększyć skórkę.

SKINHEIGHT 16 -3 10 10 SKINHEIGHT Rozmiar (wysokość) (Odstęp od krawędzi okna) (Pozycja tekstu) (Pozycja tekstu belki ekranowej)

2. Preset MUI

Preferencje MUI dostarczają mnogość opcji do zmian za pomocą myszki. Zapisanie presetu MUI jako *MojaSkora.prefs* bądź *mui.prefs* w katalogu ze skórką spowoduje, że będzie dostępny do wyboru w Preferencjach MUI. Przy ustawianiu kolorów warto zwrócić uwagę na to, że kolory palety systemu są dostępne do

wykorzystania przez MUI. Chcąc aby preset MUI był jednakowy bez względu na paletę ustawioną przez *SkinConfig* – nie należy korzystać z predefiniowanych kolorów w MUI.

Pliki graficzne MUI: *SYS:Prefs/MUIGfx/MojaSkora/MojStyl/*, *SYS:Prefs/MUIGfx/Frames/MojStyl/* oraz systemowe w *MOSSYS*: Rozszerzenie mim dla dwustanowych obrazków oraz mbr dla jednostanowych.

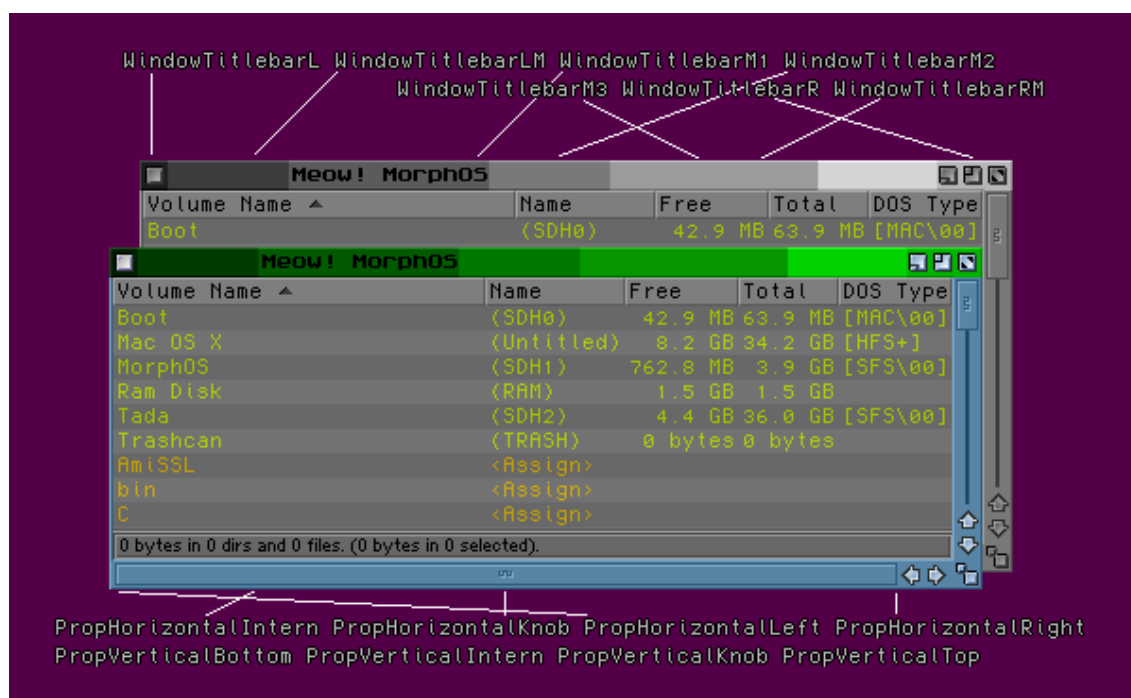
3. Obrazki modułów ekranowych

SYS:Data/Screenbar/ oraz systemowe w *MOSSYS*:

Polecam:

PPaint, *TVPaint*, *FXPaint*, *ShowCase* (dawny *ShowGirls*), *optipng*.

Malowanie czas zacząć!



Zrzut 3 - Umieszczenie obrazków okna

Gdzie MorphOS nie da rady, tam Mac OS pomoże

trOLLO

Posiadam Power Maca więc naturalne było, że oprócz MorphOS-a zainstaluję także Mac OS X 10.5.8. MorphOS może i jest doskonałym „amigowym” systemem, jednak jak każdy system NG cierpi na poważny brak dużej ilości oprogramowania. Mac OS pomaga rozwiązać wszelkie braki programowe, jakie napotka się używając MorphOS-a.

Oczywiście trzeba pamiętać, że programy, które opiszę poniżej są przydatne dla mnie i niekoniecznie ten wybór może Wam pasować, możecie wręcz uznać, że jest to całkowicie bezsensowny dobór oprogramowania.

A więc zaczynamy....

INTERNET

W dzisiejszych czasach komputer NG bez dostępu do internetu jest w zasadzie nic nie wart. MorphOS ma przeglądarkę Odyssey, która niestety od kilku lat nie była aktualizowana, co powoduje sytuację, że niektóre strony nie działają prawidłowo (np. logowanie do Allegro), lub nie działają w ogóle (np. Google Docs). Mac OS X 10.5.8 – czyli najnowszy jaki można zainstalować na komputerach Apple z PowerPC – posiada wbudowaną przeglądarkę Safari, która ma już przynajmniej 7 lat i nie potrafi wyświetlić

poprawnie większości stron. Wybór oryginalnego Safari spowoduje, że zdecydowana większość stron nie zadziała. Ratunkiem mogą być dwa popularne zamienniki starego Safari:

- TenFourFox
- Leopard-WebKit

TenFourFox to program oparty na źródłach Mozilla Firefox, obecnie najnowsza wersja to Feature Parity Release 5.

Z tą przeglądarką nie ma w zasadzie żadnych problemów. Strony, które lubię przeglądać (www.ppa.pl, benchmark.pl, expertzone.microsoft.com itd.) działają idealnie.

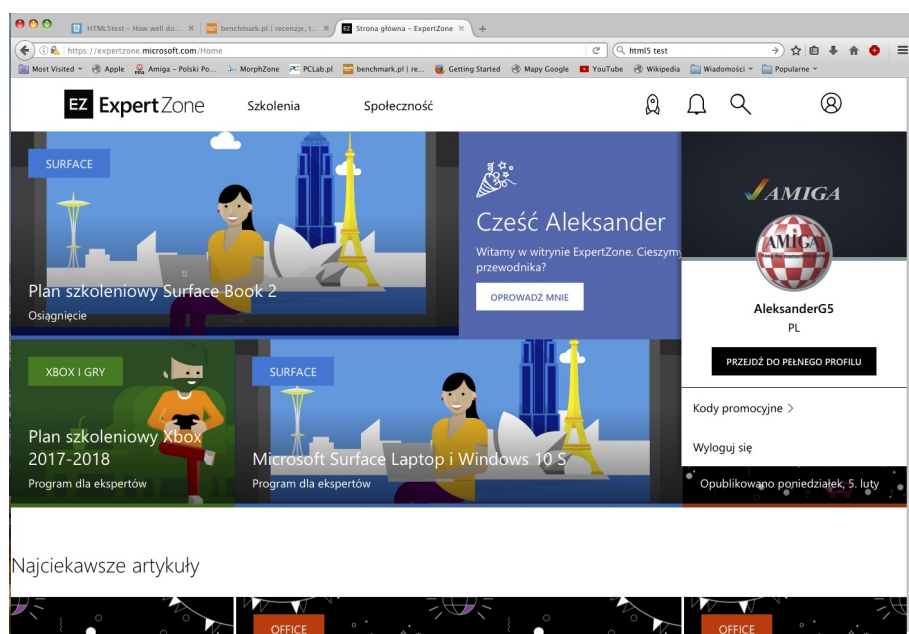
Można bez problemu logować się do Allegro, używać Google Docs.

Jak dla mnie, jedyną wadą tej przeglądarki jest jej powolne przewijanie ekranu, oraz czasami dosyć długi czas wczytywania stron www. Ogólnie, przeglądarka sprawia wrażenie strasznie ociężałej (na Power Mac Dual G5 2.0 GHz z 4 GB RAM).

Program TenFourFox można pobrać za darmo ze strony:

www.tenfourfox.com

Alternatywą dla TenFourFox jest Leopard-WebKit (w chwili pisania tekstu jest to wersja Beta 604.5.6).



TenfourFox

Program ten podmienia ustawienia Safari zawarte w systemie.

Instalacja jest prosta, ściągamy archiwum, otwieramy je, przegrywamy program WebKit w dowolne miejsce i tyle.

Zaletą Leopard WebKit jest prędkość działania. Program wczytuje się zdecydowanie szybciej od TenFourFoxa, szybciej i płynniej przewija strony www. Zabiera także mniej pamięci oraz w mniejszym stopniu obciąża CPU.

Do wad programu można zaliczyć trochę mniejszą stabilność, Leopard-WebKit potrafi się nieestety czasami zawiesić.

Program Leopard-WebKit można pobrać ze strony:

sourceforge.net/projects/leopard-webkit/files/

Oczywiście muszę przyznać, że większość stron przeglądam jednak pod MorphOS i Odyssey,

ponieważ, w moim odczuciu, Odyssey szybciej wczytuje strony www oraz szybciej przewija strony.

E-MAIL

W przypadku e-maili, pod systemem Mac OS używam Thunderbird 3.1.20. Program stary, ale doskonale spełnia swoje zadanie. Można spokojnie pisać/odbierać maile. Program bez problemu obsługuje skrzynki w kontach outlook.com, poczta.onet.pl (w tych dwóch serwisach mam skrzynki).

BIURO

Internet internetem, ale czasami pracować trzeba.

Niestety dla mnie osobiście procesor tekstu jest niezwykle ważny ze względów zawodowych – Word oraz Excel to podstawa, a tutaj MorphOS niestety nie ma nic do zaoferowania oprócz wspomnia-

nego już Folio (oczywiście są stare WordWorth7, FinalWriter itd., ale niestety nie nadają się w dzisiejszych czasach do pracy zawodowej, głównie poprzez niekompatybilny format zapisu dokumentu).

Z ratunkiem przychodzi Mac OS, który posiada wiele alternatyw.

Najpopularniejsze pakiety dostępne pod systemem Mac OS X 10.5.8 to:

- Microsoft Office 2004/2008 – to są oczywiście programy płatne
- Apple Pages z pakietu iWork'09 – ten pakiet do niedawna też był płatny
- Open Office 3.2.0
- Libre Office 4.0.6.2

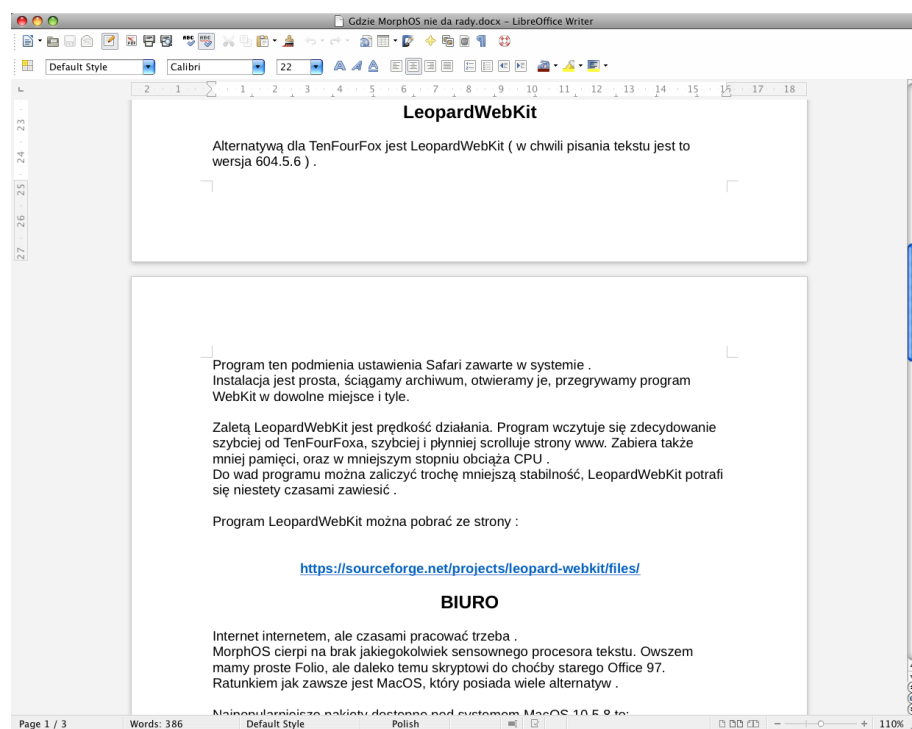
Osobiście używam Libre Office – jest najszybszy, najnowszy z dostępnych na PowerPC (wersja z 2013 r.), pozwala on też otwierać dokumenty DOCX z MS Office 2010 czy nawet Office 365; także Excel i jego XLSX nie są wyzwaniem dla LibreOffice, dzięki czemu nie mam problemów z przenoszeniem dokumentów między różnymi komputerami.

Jedynym mankamentem jest PowerPoint, no ale w wersji Office 365 PowerPoint został wzbogacony o tak wiele funkcji, że LibreOffice z 2013 r. nie może mieć możliwości odczytu takiej prezentacji.

Program można pobrać ze strony www:

downloadarchive.documentfoundation.org/libreoffice/old/4.0.6.2/macos/ppc/

Dodatkowo każdy z wymienionych programów bez najmniejszego problemu współpracuje z



LibreOffice

moją drukarką WiFi HP DeskJet F4580, pod Mac OS mogę korzystać ze skanera, drukować itd., czego niestety nie mogę zrobić w systemie MorphOS.

GRAFIKA/FILMY

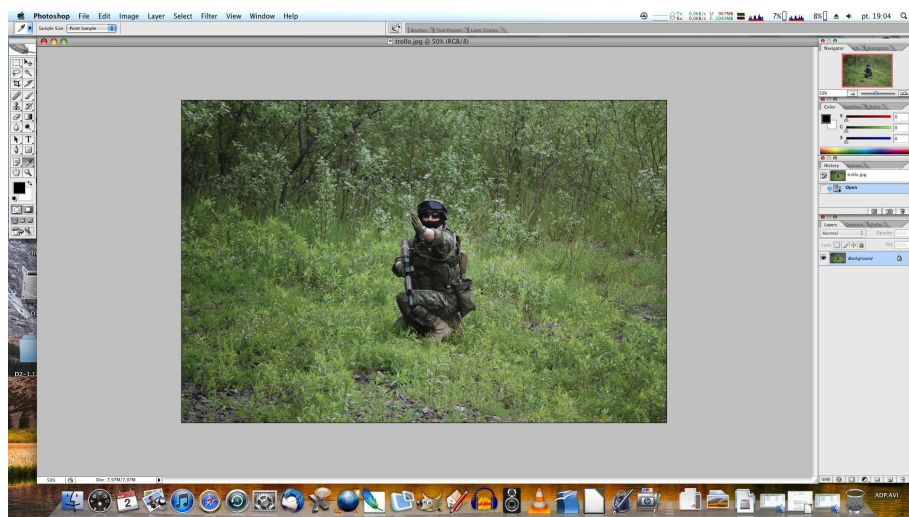
Do różnego rodzaju operacji graficznych używam GIMP-a w wersji 2.6. Trzeba tylko pamiętać, że GIMP w wersji dla Mac OS PowerPC wymaga zainstalowania środowiska X11. Bez tego niestety się nie uruchomi.

Ostatnio (bodajże w zeszłym roku) Adobe udostępnił całkowicie za darmo Adobe Photoshop CS2, wystarczy tylko zarejestrować się na stronie Adobe i już można się cieszyć tym programem.

Do oglądania filmów używam VLC 0.9.10. Program, mimo iż stary, bez problemu odczytuje wszystkie filmy jakie mam na dysku. Oczywiście ze względu na moc obliczeniową PowerPC G5, nie ma co liczyć, aby komputer odtworzył modne ostatnio filmy 4K.

MUZYKA

Programów do odtwarzania plików MP3/FLAC na Mac OS PowerPC jest bardzo dużo, ale najprostszym jest Vox 0.2.7.1. Program jest bardzo mały, prosty i nie obciąża komputera. Wadą programu jest fakt, że odtwarza tylko MP3/OGG/FLAC, no ale, jako że większość mojej kolekcji to MP3/FLAC, nie przeszkadza mi fakt, iż Vox nie rozpoznaje innych formatów.



Photoshop



Vox

INNE

Wyżej wymienione programy to te, które używam najczęściej, jednak to nie wszystkie. Jest jeszcze Google SketchUp, Audacity, HP Utility (skaner, drukarka), X-Chat Aqua, Virtual PC, Blender; no i oczywiście gry: Red Alert 3, Halo 1, Command & Conquer: Generals.

Pominałem też całkowicie programy do edycji i montażu filmów jak np. FinalCut 6, programy typu CAD itd. Jak pisałem na początku artykułu, każdy z nas ma inne wymagania i dla każdego z nas wybór niezbędnych programów będzie inny.

Dlaczego tak?

Moim zdaniem, MorphOS i Mac OS idealnie się uzupełniają. Często mam uruchomionego MorphOS-a, ponieważ w danej chwili wykonuję podstawowe czynności takie jak:

- słuchanie muzyki
- oglądanie filmów
- rozmowa na IRC-u, itd.

Jednak jak przychodzi potrzeba napisania jakiegoś tekstu, poprawy jakiejś tabelki do pracy, to wtedy przełączam się na Mac OS, ponieważ tam mam programy, które są mi potrzebne do wykonania danego zadania. Owszem, mam nowoczesnego iMaca z Core i5, 16 GB RAM, jednak MorphOS i czasami Mac OS X 10.5.8, idealnie nadają się do pracy i luźnego spędzania czasu przed komputerem.

Instalacja AROS-a w maszynie wirtualnej

Don Rafito

Nie ma to, jak zacząć od tyłu. AROS to, AROS tamto, taka, gra, siaka gra. A tu trzeba było zacząć od prezentacji i omówienia instalacji systemu, który jest jednym z przedstawicieli Amigi NG. Wybaczcie mi ten błąd, ale co zrobić, taka już nasza tradycja narodowa, którą wprowadzały kolejne nacje polityczne, próbując dotrzeć do problemów rodaków przez okienko w składziku na narzędzia.

AROS to początkowo Amiga Research Operating System, ale ze względu na brak zgody na wykorzystanie słowa „Amiga”, stało na obecnej nazwie; to system bazujący na źródłach AmigaOS 3.1 i dążący do kompatybilności z nim, ale o wiele bardziej go obecnie przewyższający. Stworzony jeszcze w latach 90-tych, na fundamentach systemu Linux, przeznaczony był głównie dla komputerów klasy PC, czyli z procesorami o architekturze x86. Obecnie są już kompilacje, które można odpalić na procesorach PowePC, m68k czy ARM. Ten odstani występuje także w wersji dla systemu Android i można go odpalić np. na smartfonie w trybie HOSTA, jako osobny proces, przy czym nie jest to typowa emulacja. Po więcej szczegółów odsyłam Was do sieci (np. www.aros.org), a my bardziej skupimy się na tym, jak AROS-a zainstalować i uru-

chomić na swoim komputerze.

Zdecydowanie polecam instalację systemu na natywnym sprzęcie klasy właśnie x86, czyli najpowszechniej dostępnym, który dzięki odpowiednio dobranym podzespołom i sterownikom, najefektywniej będzie wykorzystany przez nasz postamigowy system. Aby dowiedzieć się więcej na ten temat, proponuję stronę:

en.wikibooks.org/wiki/Aros/Platform/x86_support

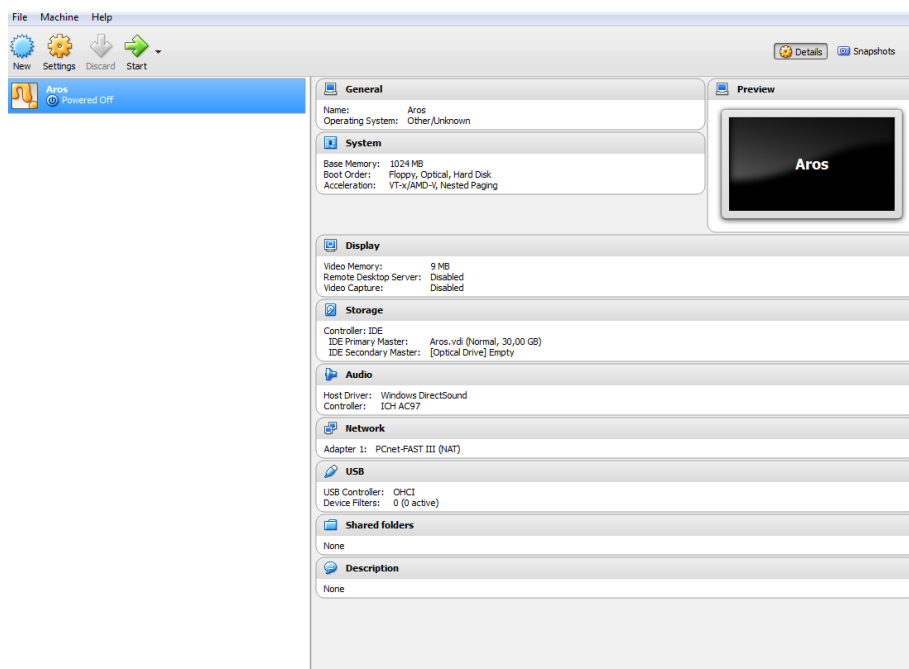
Tam znajdziecie stosowne informacje, dzięki którym dobór sprzętu będzie optymalny. Niemniej jednak większość zapewne wolałaby najpierw przetestować OS, zanim dojdzie do ewentualnego zakupu hardware, dlatego skupimy się na odpaleniu AROS-a w programie typu maszyna wirtualna.

Na rynku, jest kilka firm oferujących oprogramowanie, służące emulacji innych platform sprzętowych. Jedne maszyny są płatne inne darmowe. Jaką wybierzeć aplikację, zależy od Waszego gustu i nie będziemy zbyt skłaniać się nad jej doбором i szczegółowego omawiać każdej z nich. Po tę wiedzę odsyłam Was do pism komputerowych traktujących o platformach PC. Osobiście, bazując na swojej skromniutkiej wiedzy i doświadczeniu, wybrałem program *Virtual Box*, w wersji portable i zaprezen-

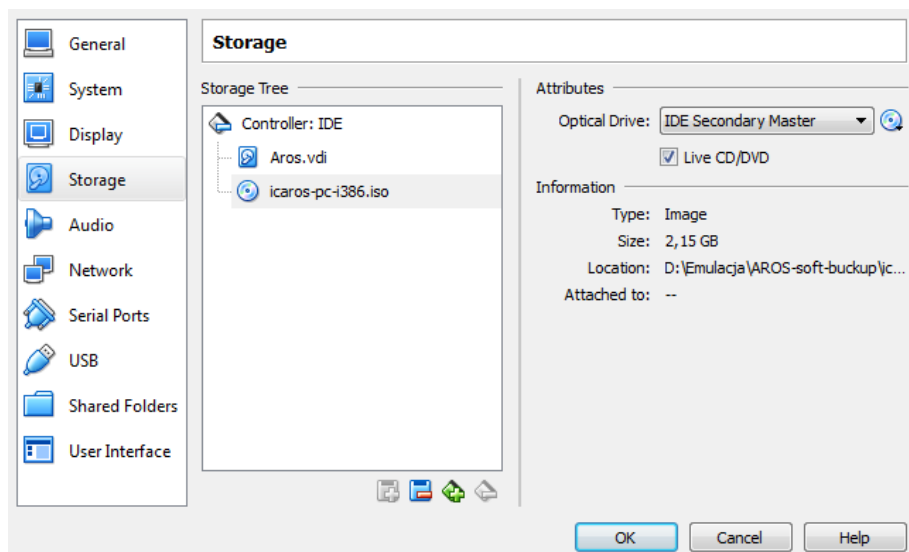
tuję jak stworzyć sprzęt potrzebny do uruchomienia i zainstalowania systemu AROS.

Po pobraniu i zainstalowaniu maszyny przechodzimy do „zbudowania” komputera, na którym zainstalujemy nasz omawiany system. Odpalamy program, klikamy w **New**, w nazwie wpisujemy **Aros**, a w **Type** i **Version** wybieramy **Unknown**. Klikamy w **Next** i przechodzimy do ustalenia ilości pamięci dla systemu. 1GB w zupełności wystarczy, zatem ustawiamy na 1024 MB i klikamy **Next**. W następnym oknie mamy opcje kreowania dysku twardego. Wybieramy **Create a Virtual Disk Now** i klikamy w **Create**. Jako typ wybieramy format **VDI**, w następnym oknie wybieramy alokację dynamiczną, dzięki czemu plik będzie się nam powoli rozszerzał, do ustalonej wielkości, w miarę zapewniania go, a nie zajmie od razu całej zadeklarowanej przestrzeni. W następnym oknie dobieramy rozmiar. Spokojnie wystarczy tu ze 30 GB, które potem podzielimy na partycje już podczas instalacji AROS-a, przeznaczając 10 GB na system, a pozostałe 20 GB na dodatkowe oprogramowanie. Klikamy w **Create**. Jeśli wszystko wykonaliśmy prawidłowo, mamy już przygotowane podwaliny po nasz postamigowy system (*Zrzut 1*). Aby doszlifować niektóre opcje,

klikamy w **Settings** i przechodzimy do zakładki **Display**. Od razu wyjaśniam; w maszynie wirtualnej, obojętnie którego producenta, nie uzyskamy efektu dedykowanej karty graficznej. Podczas instalacji natywnej na sprzęcie z grafiką Intel GMA, czy NVidia GF (w obu przypadkach, też nie wszystkie modele), będzie taka możliwość, ale po szczegóły odsyłam Was do wspomnianej powyżej strony suportu na *Wiki-books*. Na emulatorze, zawsze będzie to tylko tryb VESA, który dobrze sobie radzi z grafiką 2D oraz 3D, w momencie, kiedy nie jest wymagana akceleracja sprzętowa w danej aplikacji, w przeciwnym razie program/gra albo wcale się nie uruchomi, albo będzie tak wyświetlana, że przypominać to będzie raczej slide show. Co więcej, w trybie VESA, dostępny jest tylko jeden tryb w **Screen Mode Preferences**; maksymalny, obsługiwany przez nasz monitor i kartę graficzną gospodarza. Nie mniej jednak proponuję ustawić „jakiś tam” rozmiar pamięci graficznej (w moim przypadku jest to 64 MB) i włączyć **akcelerację 3D**, ale tylko **3D**. Opcji **2D** nie ruszamy. Następnie, przechodzimy to zakładki **Storage**. Klikamy w ikonkę **krażka CD**, z prawej strony będziemy mogli wskazać czy chcemy korzystać z fizycznego napędu gospodarza, czy z obrazu płyty. W sekcji **Attributes**, w **Optical Drive**, poprzez kliknięcie, ponownie w ikonkę **krażka CD** i wyboru opcji **Choose Virtual Optical Disc File...** wybieramy plik naszego obrazu płyty. W moim przypadku, postanowiłem zainstalować dystrybucję *Icaros Destop*, którą notabene mam obecnie postawioną na natywnym sprzęcie. Wchodzimy na stronę



Zrzut 1



Zrzut 2

www.icarosdesktop.org i w zakładce **Download**, znajdziemy archiwa z proponowanymi wersjami systemu AROS. Możemy pobrać systemem w wersji pełnej, zawierający preinstalowane oprogramowanie, lub w wersji Lite, zawierającej tylko OS z podstawowym oprzrządowaniem. Na wskazanej stronie, znajdziemy także patche i updaty do już zainstalowanych wcześniej, starszych

wersji *Icaros Desktop*. Proponuję pobrać najnowszą wersję, z pełną opcją oprogramowania. Po pobraniu i rozpakowaniu archiwum, wybraniu pliku **icaros-pc-i386.iso**, ptaszkujemy jeszcze opcję **Live-CD/DVD** (Zrzut 2).

Ostatnią opcją w ustawieniach naszej maszyny wirtualnej, której będziemy się tykać, to zakładka **Audio**. Może zdarzyć się tak, że w

późniejszym etapie, już po zainstalowaniu systemu, nie będziemy mieć dźwięku, mimo prawidłowo rozpoznanej karty. Zatem proponuję, w owej zakładce, w **Host Audio Driver**, przestawić tryb na **Windows Direct Sound**. W moim przypadku zadziałało. Nawet jeśli nie zrobimy tego teraz, zawsze możemy jeszcze doszlifować niektóre opcje, już po zainstalowaniu AROS-a. Klikamy **OK**, na głównej stronie programu i uruchamiamy nasz postamigowy system. Podczas startu przywita nas *GRUB*, menu znane z Linuxa, w którym możemy przebierać w opcjach wyboru różnych parametrów startu OS-u, ale niczego tu nie ruszamy, gdyż *Icaros* sam dobierze optymalne ustawienia. Po chwili ukaże się nam blat, znajomo zatytułowany *Workbench Screen* z nakładką *Opus Magellan*, z napędami i skrótami. Jak widać, AROS załadował się w oknie, ale bez problemu możemy migrować w *VirtualBoxie*. do opcji **Full Screen**, poprzez kombinację klawiszy **Host+F** (klawisz Host, jest tu zdefiniowany pod **prawym Ctrl**). Bez ogródek odpalamy instalator (**InstallAROS**), w pierwszym oknie klikamy **Proceed**. W następnym wybieramy

opcję **Wipe disk**, dzięki czemu, przejdziemy do funkcji podziału dysku na partycje. Tak jak wspomniałem wyżej, 10 GB przeznaczamy na system, pozostałe 20 GB, na dodatkowe oprogramowanie (*Zrzut 3*).

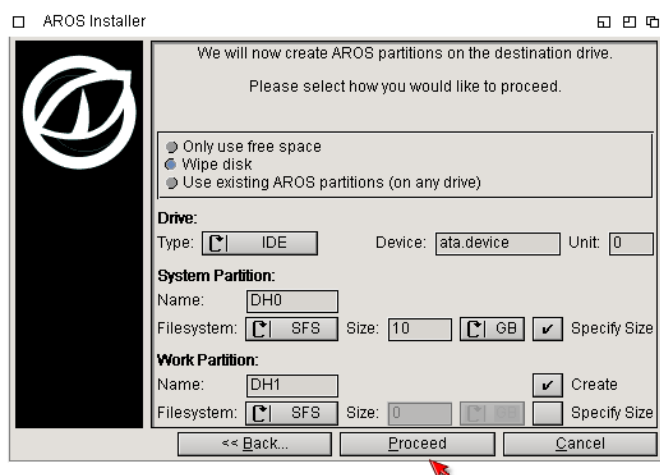
Klikamy **Proceed**, a następnie będziemy poproszeni o reset systemu. Na blacie pojawią się nasze partycje. Znowu uruchamiamy instalator, **Proceed** i tam następnym oknie, gdzie ustawialiśmy rozmiar partycji, zaznaczamy trzecią opcję **Use Existing AROS partitions (on any drive)** i klikamy **Proceed**. W następnym oknie zaznaczamy, co chcemy mieć w systemie zainstalowanie. Ja zaznaczam wszystko, prócz oprogramowania dla developerów i klikamy **Proceed** (*Zrzut 4*).

W następnym oknie zaznaczamy wszystko, dzięki czemu dodatkowe oprogramowanie zostanie zainstalowane na partycji **Work**. I tradycyjnie **Proceed** (*Zrzut 5*). Potem **Proceed** razy trzy (w międzyczasie będzie będziemy pytani gdzie chcemy mieć bootloader *GRUB* i czy chcemy go w postaci graficznej, nic nie zmieniamy) i w ten oto sposób przejdziemy do

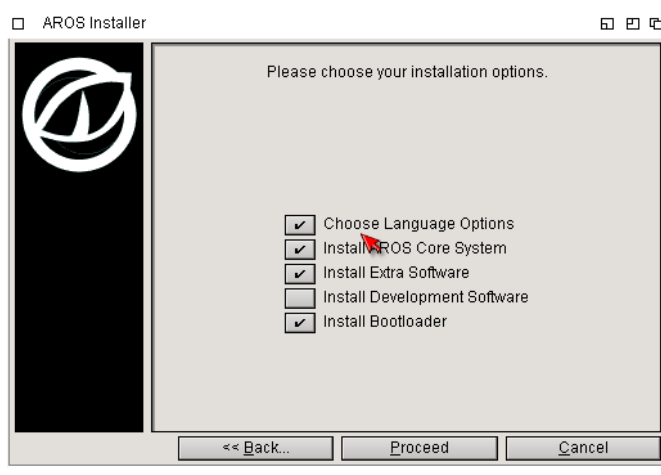
opcji językowych. W **Input Preferences**, a następnie w **Region Language** szukamy **Polska** i potwierdzamy w obu przypadkach klikając **Use** (*Zrzut 6*). Po wszystkim idziemy na kawę/piwko, kto co woli (*Zrzut 7*).

Po zainstalowaniu oprogramowania systemowego, przechodzimy do wyboru oprogramowania dodatkowego. Tu nie ma co przebierać i zaznaczamy **Full Installation** i idziemy na drugą kawę/browar. Instalatorowi troszkę zajmie rozpakowywanie archiwów, więc nie ma się co spieszyć (*Zrzut 8*).

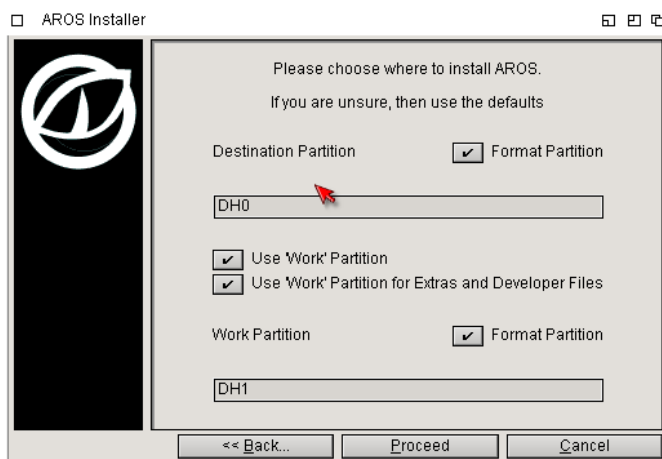
W trakcie wyskoczy nam jeszcze powiadomienie o aktualizacji **AISS**, którą potwierdzamy i instalujemy oprogramowanie dodatkowe dalej. Po zakończeniu instalacji poproszeni jesteśmy o wysunięcie płyty instalacyjnej (obrazu iso) z napędu, a następnie wykonania resetu komputera. Wyłączamy emulację (**Host+F**, jeśli działaliśmy na pełnym ekranie) zamykamy okno odpalonego AROS-a (potwierdzając opcję **Power Off...**) i klikając w **Settings**, przechodzimy do **Storage**. W sekcji **Attributes**, **Optical Drive**, klikamy na symbol płyty i



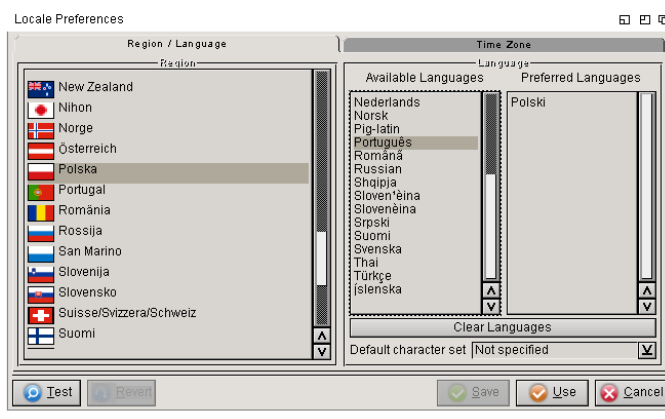
Zrzut 3



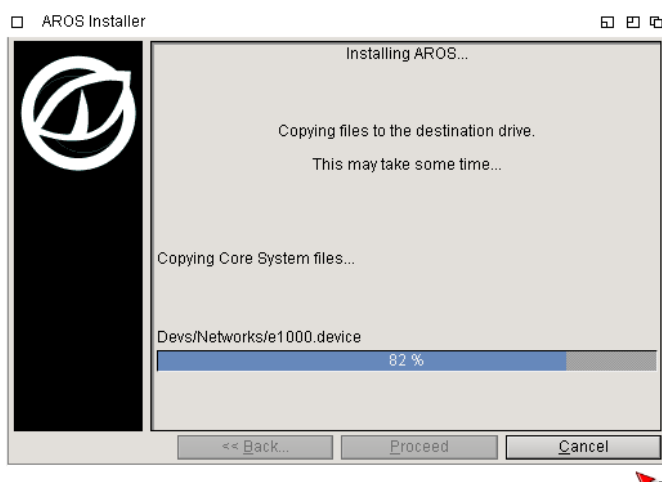
Zrzut 4



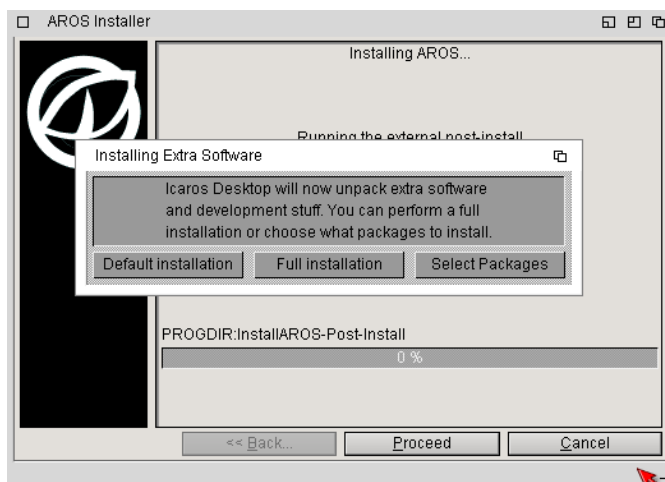
Zrzut 5



Zrzut 6



Zrzut 7



Zrzut 8

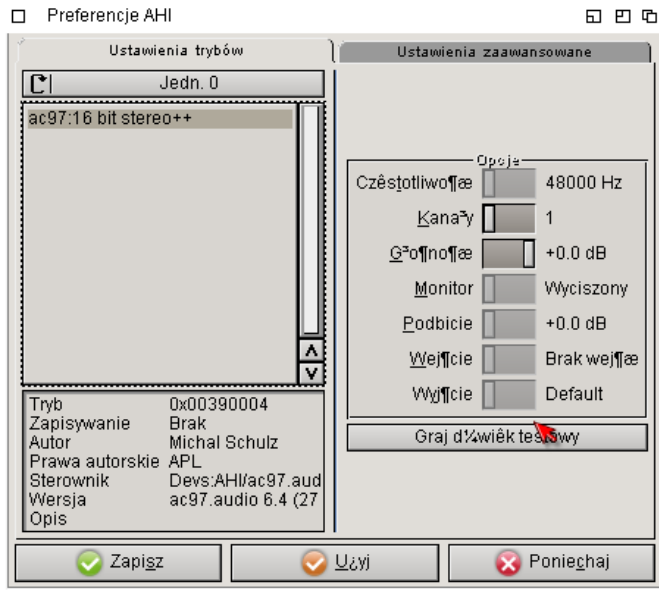
wyberamy **Remove Disk from Virtual Drive**. Klikamy **OK**, a następnie starujemy *Icarosa*. Po przejściu przez *GRUB-a*, zostajemy poinformowani o pierwszym użyciu i system proponuje nam wprowadzenie pierwszych ustawień. Klikamy **YES** i przechodzimy do opcji językowych i lokalizacyjnych. W tym przypadku klikamy na **Zapisz**. W następnym oknie, w **Input Preferences**, wyszukujemy **Polska** i także klikamy **Zapisz**. W **Screen Mode Preferences**, wybieramy nasz tryb **VESA**; jak wspomniałem, do wyboru będzie tylko jeden i znów klikamy w **Zapisz**, niczego więcej nie zmieniając. Po przejściu do ustawień **AHI**. Jak widać, AROS

rozpoznał naszą kartę dźwiękową, w tym przypadku jest to **AC97**. Zaznaczamy ją, a następnie z prawej strony, suwak **Głośność** przestawiamy na maksa. Potem klikamy na menu rozwijane zatytułowane **Jednostka Muzyczna** i wybieramy **Jedn.0**. Znowu wybieramy naszą kartę i zaś suwak głośności przestawiamy na **+0.0 dB**. Dla pewności klikamy w **Graj dźwięk testowy** i słuchamy, czy z głośników płynie dźwięk. Jeśli tak klikamy **Zapisz** i jedziemy dalej. Może być tak, że tego dźwięku nie będzie, wtedy trzeba już pogrzebać w ustawieniach *VirtualBoxa*, o których wspominałem w konfiguracji naszej platformy pod *Icarosa* (Zrzut 9).

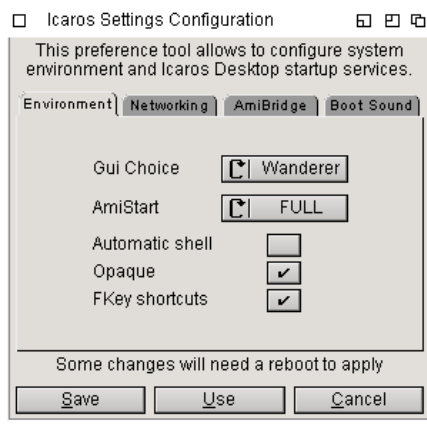
W **IControl**, klikamy **Poniechaj** (opcje standardowe uważam za dobre) i przechodzimy do okna **Icaros Settings Configuration**. Nie jestem zwolennikiem *Magellana*, to też w zakładce **Environment** wybrałem opcję **Wanderer**, a *AmiStart* przestawiłem na **FULL**. Pozostałych opcji radzę nie ruszać na razie. Np. opcją **AmiBridge**, zajmiemy się następnym razem (Zrzut 10).

Klikamy na **Save** i proszę, *Icaros Desktop* w pełnej krasie (Zrzut 11).

Jak zauważyliście, podczas wstępnej konfiguracji systemu, niektóre polskie napisy pojawiają się z krzaczkami. Zanim zaczniemy



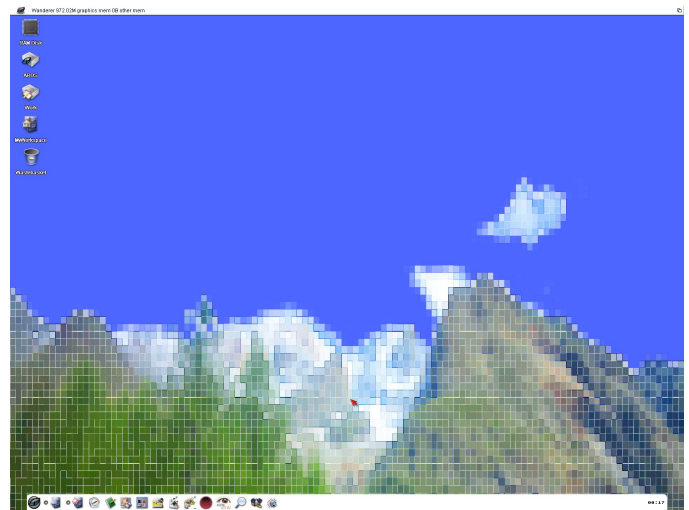
Zrzut 9



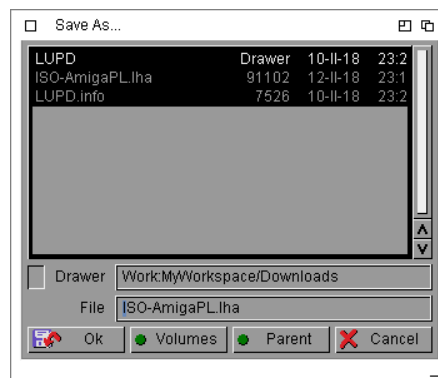
Zrzut 10

zabawę z *Icarosem*, proponuję jeszcze tę sprawę uregulować. Na dolnym pasku *AmiStart* uruchamiamy *OWB* (przy okazji zobaczymy sobie jak śmiga internet w naszym systemie) i wchodzimy na *Aminet.net*. Szukamy czcionki **ISO-AmigaPL**. Zapisujemy sobie w **Work:MyWorkspace/Downloads** (Zrzut 12).

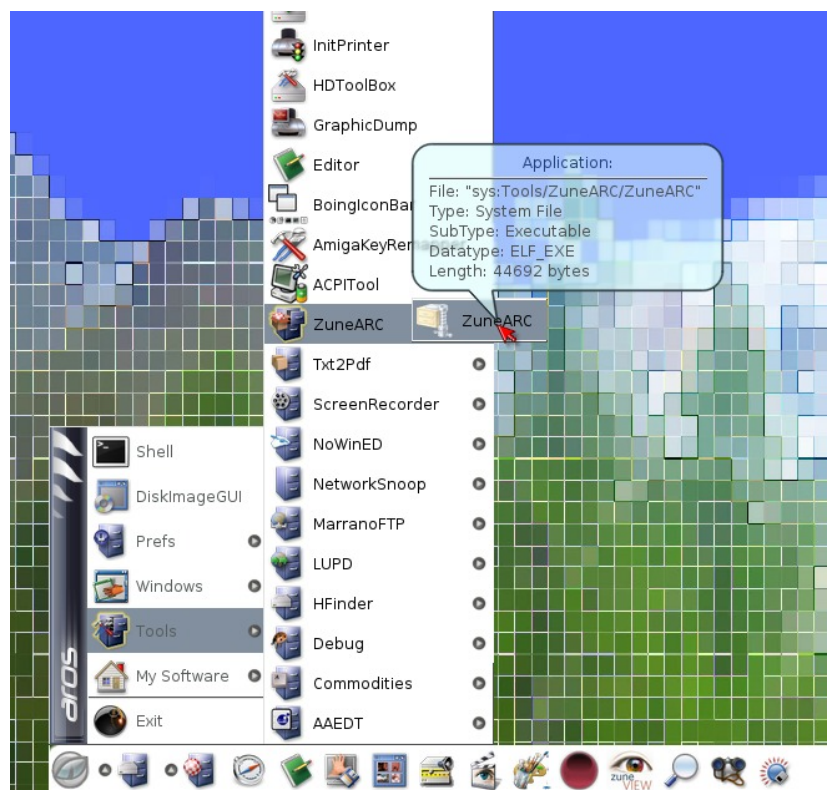
Klikamy na logo AROS-a na pasku *AmiStart*, potem *Tools/Zune-ARC/ZuneArch* i uruchamiamy archiwizator (Zrzut 13).



Zrzut 11



Zrzut 12

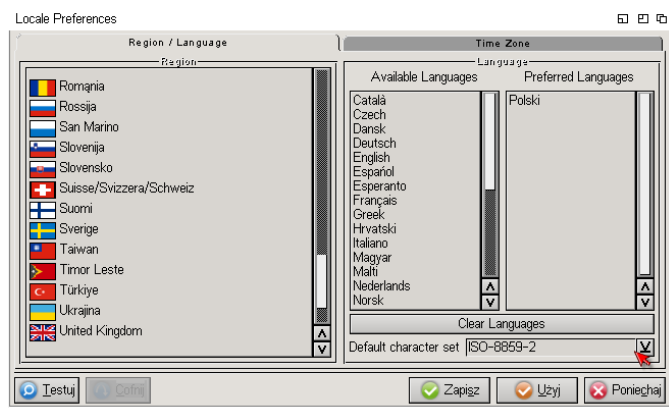


Zrzut 13

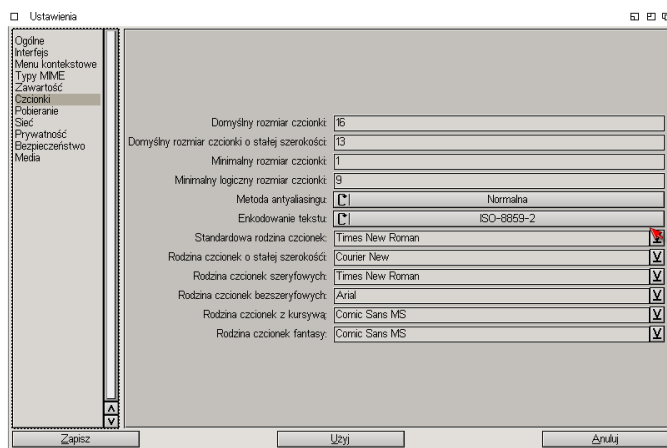
OK Archivers configuration

```
CmdName ,LHA,
ExtractSingle ,#, lha "-xw=%dest" %arch %file,
ExtractAll ,stack 64000;C:lha "-xw=%dest" %arch,
ListContent ,stack 64000;C:lha -v %arch,
ContentHeader ,2,
ContentFooter ,2,
ContentFmt ,OTHE OT10 CSIZ SIZE OTHE OTHE OTHE DATE DATE NAME,
Add ,stack 64000;cd %base;C:lha -a %arch %file,
Remove ,#, C:lha -d %arch %file,
Create ,C:touch %arch,
```

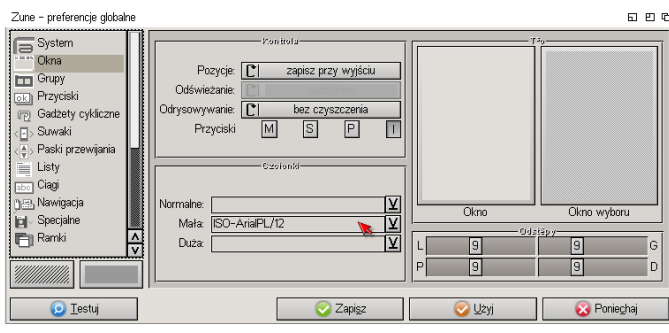
Kod 1



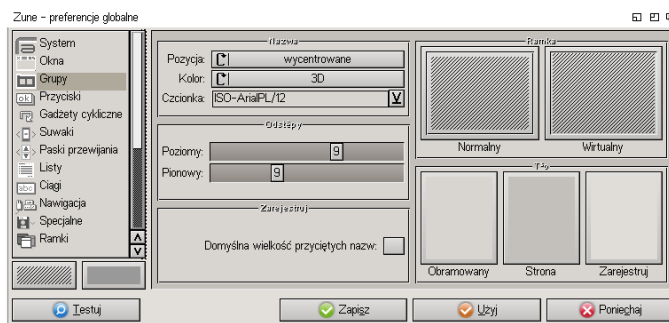
Zrzut 17



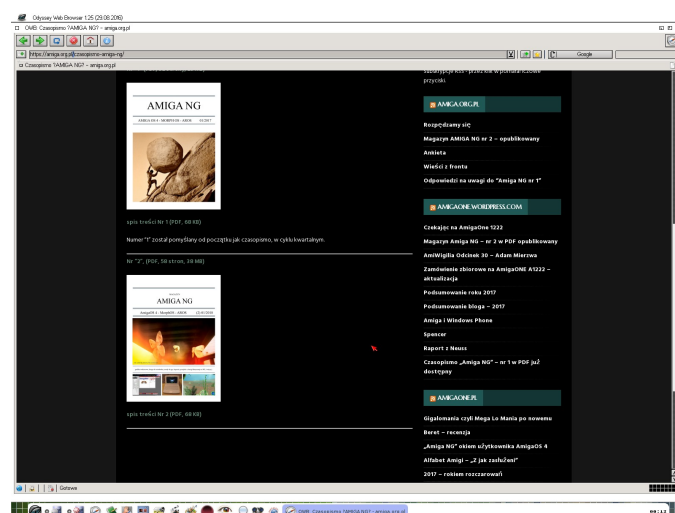
Zrzut 20



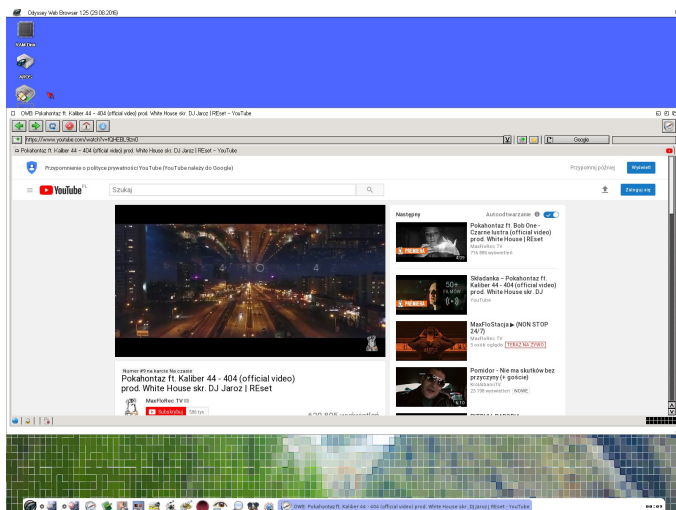
Zrzut 18



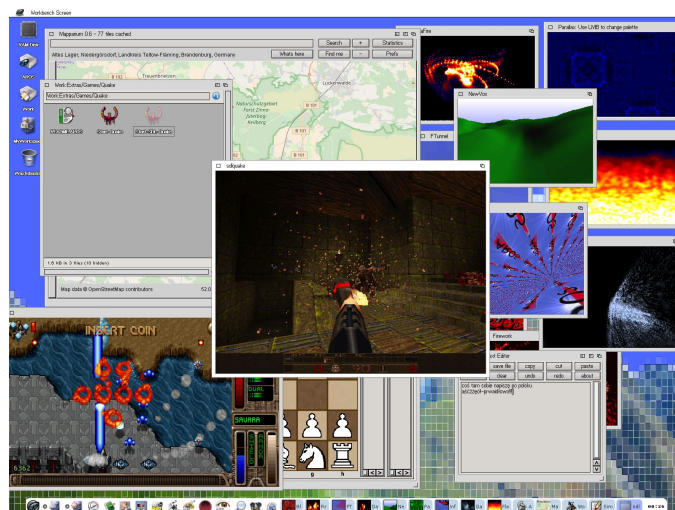
Zrzut 19



Zrzut 21



Zrzut 22



Zrzut 23

REKLAMA

K&A plus

Magazyn użytkowników komputerów Commodore

Z pasją o kulturze retro gier komputerowych:

- recenzje
- felietony
- tutoriale
- dyskusje
- demoscena
- sztuka
- programowanie
- wywiady
- emulacja
- nowości

Komoda & Amiga plus to pismo poświęcone szerokokopijnej kulturze retro gier komputerowych. Mimo że nasze rozważania poświęcone są głównie produkcjom publikowanym na sprzęt spod znaku marki Commodore to jednak celem nadrzędnym jest odświeżenie tego niesamowitego klimatu lat 80/90-tych, czyli czasów naszego dzieciństwa wypełnionego pikselami i dźwiękiem chipowym.

www.ka-plus.pl

team@ka-plus.pl

Amiga 68k, AmigaOS 4.x, AROS, MorphOS

Plus/4, VIC20, C16, C64, C128

Hermes

Don Rafito



Konsumpcja żywności to jedna z podstawowych potrzeb nie tylko człowieka, ale i wszystkich żywych organizmów. Tylko człowiek zrobił w tej dziedzinie najwięcej. W dzisiejszych czasach produkcja i spożywanie żywności zostały połączone już chyba ze wszystkimi sferami życia. Od prostych posiłków codziennego dnia, celebrowanie spotkań przy stole, poprzez masowe imprezy, reklamy, supermarkety, fast foody, na wykwintnych restauracjach, daniach będących produktem sztuki kulinarnej i proekologicznych trendach, kończąc.

Skoro tak, to dlaczego tematu jedzenia nie wcisnąć by w świat gier komputerowych? Historia roz-

rywki elektronicznej pokazała, że było to już robione tak jak w przypadku naszego dzisiejszego tytułu. Hermes, ewidentnie nawiązuje do wytwarzania żywności i jej spożywania. Tytułowy bohater, kucharz, miał chyba dość niezdrowego i szybkiego jedzenia, a że jest niezłym obżartuchem, o czym będzie można się przekonać podczas rozgrywki, postanowił zaspokoić kolejną swoją kulinarną zachciankę: grillowanego kurczaka.

Hermes pochodzi z Francji, kraju, w którym do gotowania podchodzi się jak do sztuki. Więc jak na Francuza przystało, nasz przyjemniaczek postawił na rozwiązanie eko i zamiast iść do marketu i

kupić gotowe mięso, postawił na żywy inwentarz, z wolnego wybiegu. A nasz wybieg jest bardzo duży, bo liczy aż 32 poziomy.



Retroguru, grupa słynąca z takich tytułów jak Sqrzx, czy Giana's Return, tym razem stworzyła platformówkę, oczywiście utrzymaną w klimacie retrogamingu, ze sporą dawką czarnego humoru. Jako gracze, wcielamy się w rolę naszego Hermesa i z tasakiem w rękę, próbujemy dorwać latający po planszach żywy drób. Nie jest to zadanie proste, a kurak jest dość szybki. Na naszej drodze staną nam, nie tylko przeszkody architektoniczne, najróżniejszej maści maszyny pozbawiające nas życia, rozmaite stworki (niektóre z nich można eliminować poprzez tradycyjny skok na głowę), ale także samo obżarstwo głównej postaci. Hermes wraz z postępem pościgu, staje się coraz bardziej głodny. Na jego szczęście po drodze rozłożone są różne formy niezdrowej, szyb-

kiej żywności, np. pączki. Ale uwaga, przejedzenie grozi nagłym niekontrolowanym oddaniem kupy i utratą życia.

Grupa nie raz pokazała, że potrafi zrobić przyzwoitą, utrzymaną w klimacie retro, grę, z dobrą i czytelną pikselową grafiką. I tak jest w tym przypadku. Sceneria jest na dobrym poziomie, postacie i całe plansze są nieźle wykonane. Choć z drugiej strony, mogłoby powiać świeżością. Tylko czy wtedy programiści z teamu, nadal mogliby się nazywać Retroguru? Może lepiej niech zostanie jak jest, a jest całkiem przyzwoicie.

Chiptune'owe podkłady muzyczne oraz różne dźwięki i efekty specjalne towarzyszące rozgrywce, dobrze się komponują z akcją na ekranie. Całości dopełniają odgłosy, towarzyszące procesowi przetwarzania żywności; od wizyty przy stole do odwiedzin w ubikacji włącznie, co podkreśla ów czarny humor, towarzyszący całemu tytułowi.

Hermes doczekał się edycji niemalże na wszystkich wiodących platformach. Miło, że do tego grona dołączyły także postami-gowe systemy. Jeśli czasem macie dość tych wszystkich strzelanek, wyścigówek i mordobic, zapodajcie sobie Hermesa, a gwarantuję, że prócz przyzwoitej grywalności, towarzyszyć Wam będzie uśmiech i dobry humor. A na razie życzę wszystkim smacznego.



Tytuł: Hermes
Platforma: AROS, AmigaOS 4.x i MorphOS
Firma: Retroguru
Rok produkcji: 2017
Grafika: 7
Muzyka: 7
Grywalność: 8



Aspire_games_1

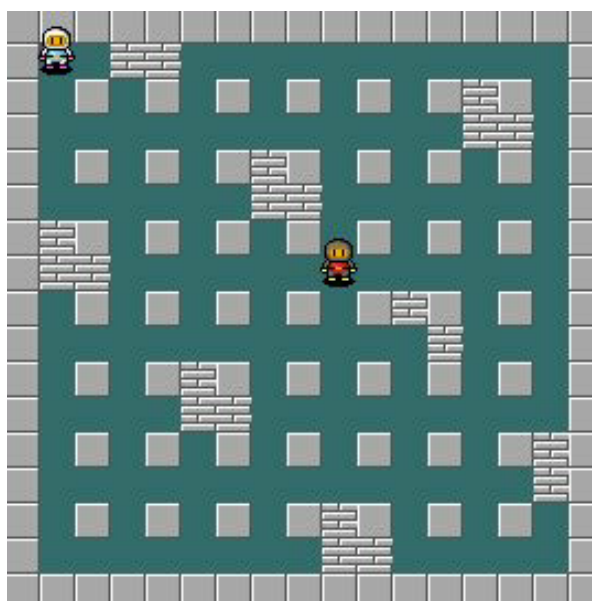
Don Rafito

Pamiętam czasy, kiedy z wypiekami na twarzy wrzucało się do napędu cover dysk w postaci dyskietki, czy płyty, z nowo zakupionego numeru ulubionego pisma komputerowego. Internet raczkował, nie wszystko było podane na tacy, czas oczekiwania na kolejne wydanie gazety z „kowerkiem”, wypełniało się katowaniem tego, co już się ma. W dzisiejszych czasach gry, użytki, dema, można kilkoma ruchami mieć już na dysku. Nie ma oczekiwania, nie ma wypieków na twarzy, za to jest pęd i pośpiech. Wszystko na wczoraj, tu i teraz. Klik, klik i już jest. Dlatego z wielką sympatią podchodzę, do wszelkich składanek, cover dysków, które się jeszcze czasami pojawiają w sieci i to nie tylko w przypadku postamigowych systemów, ale także Amigi klasycznej. No właśnie, o ile w przypadku klasyka coś się tam jeszcze od czasu do czasu dzieje w tej sprawie, to w przypadku AROS-a, MOS-a i AOS 4.x, ten proceder jest niemal znikomy. Tym bardziej, z ów wypiekami na twarzy, przysiadłem do składanki *Aspire_Games_1*.

Omawiany pakiet, liczący bagatela ponad 140 MB (he, mój pierwszy twardek w Amidze 1200 miał 120), wchodzi w skład dystrybucji AROS-a o nazwie **AspireOS**, co nie znaczy, że nie można jej używać na innej, tego postamigo-



Beret



Bomber

wego systemu. Jako użytkownik **Icaros Desktopa**, postanowiłem to sprawdzić.

Na marginesie: **AspireOS** jest dystrybucją systemu AROS właśnie, przygotowaną głównie dla komputerów Acer Aspire, ale jeśli spróbujemy ją zainstalować na innym komputerze klasy PC, którego podzespoły są obsługiwane przez ten system, oraz dostępne są sterowniki, nie powinno być większych problemów z uruchomieniem jej. Do dzieła.

Po wejściu na stronę **archives-aros.exec.org** przechodzimy do zakładki **Games** i tam znajdziemy archiwum pod nazwą **aspire_games_1.1-i386.zip**. Po pobraniu rozpakowujemy plik w dowolne miejsce. Radzę przygotować osobny katalog. Gdy już wykonamy te, powiedzmy szczerze łopatologiczne czynności, otwieramy katalog i naszym oczom ukaże się kilkadziesiąt tytułów, niewymagających żadnej dodatkowej instalacji i konfiguracji. Po prostu wchodzimy, w kolejne foldery i zarywamy kolejne noce z rzędu.

A co w środku? Oczywiście gry. Wiele z nich to mniej lub bardziej znane pozycje mające także swoje wersje na pozostałych postamigowych platformach.

Część z nich można spotkać w innych dystrybucjach AROS-a. Osobiście, dzięki swojej, zetknąłem się już z takimi tytułami jak: *Beret* – platformówka opowiadająca o historię pewnego kulistego naukowca, wcielającego się w rolę komandosa; *Bomber* – arosowy remake starej planszówki Bomberman; *Giana's Return* – nieoficjalna kontynuacja hitu



Giana's Return



Road Fighter

Giana Sisters, znanego z Commodore 64 i Amigi Classic; *Road Fighter* – remake wspaniałej samochodówki, wydanej niegdyś na platformę NES; czy cała seria *Sqrxz* – gry platformowej opowiadającej o przygodach pewnego królika.

W paczce zawarto także kilka ciekawych portów, np.: *Duke Nukem 3D* – jeden z lepszych FPS-ów z lat 90., *Flashback* – pecetowa wersja hitu na Amigę klasyczną, *Open Tyrian* – multiplatformowa strzelanka kosmiczna, *GLQuake* i *Quake III* – no tego to chyba nie muszę tłumaczyć, czy *Return to*



Sqrzx



Duke Nukem 3D



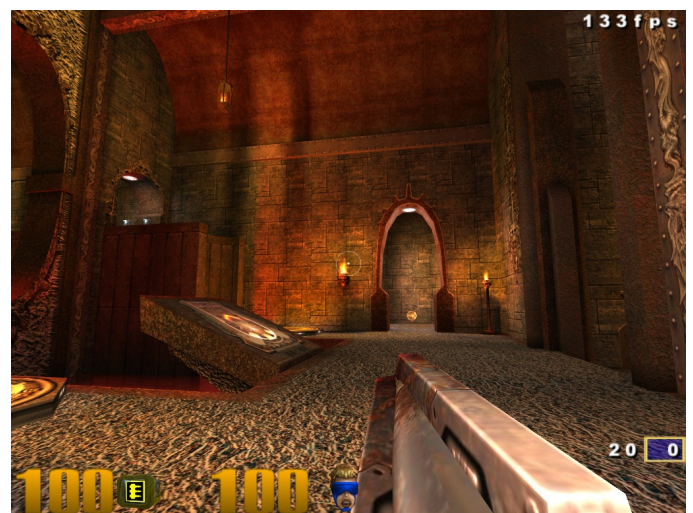
Flashback



Open Tyrian



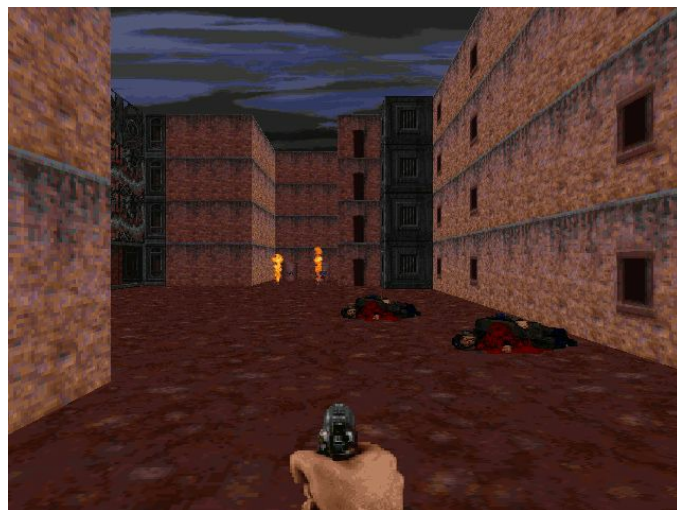
GLQuake



Quake III



Return to Castle Wolfenstein



Rott

Castle Wolfenstein – kolejny niezły pecetowy FPS.

Poza tym spotkałem się z kilkoma tytułami, których nie kojarzyłem wcześniej, ale postanowiłem je przetestować. Są to m.in.: *Dragon Memory* – prosta gra logiczna oparta na kartach memory; *Tri Mines* – dość nietypowa adaptacja znanego z pierwszych wersji systemu **Windows** tytułu Saper; *Rott* – a właściwie *Rise of The Triad*, zbudowana na bazie silnika gry Wolfenstein 3D i wydana 1994 strzelanka 3D, tutaj jako port, czy *XRick* – klon gry Rick Dangerous, platformówki znanej z Commodore 64. Wszystkich tytułów nie wymieniałem, aczkolwiek każdemu z nich należy się chwila uwagi. Na pewno każdy znajdzie coś dla siebie.

Tak jak wspomniałem wyżej, wiele z nich można spotkać w innych dystrybucjach, lub zwyczajnie poszukać w sieci, choćby na wspomnianej na początku stronie internetowej, lub innych skupiających oprogramowanie dla pozostałych postamigowych systemów.



Dragon Memory

Pora na minusy. Najbardziej razi fakt, że autor projektu, zwyczajnie, przepraszam za wyrażenie, zerznął cały pakiet gier i zamknął je w jednym archiwum. Ja rozumiem, że prawie wszystkie są wersjami darmowymi, pomijając niektóre porty, do których uruchomienia potrzebne będą pliki z oryginalnych tytułów i które mogą być objęte prawami autorskimi, o czym autor poinformował potencjalnego gracza. Rozumiem też, że ów pakiet wchodzi w skład wspomnianej dystrybucji AROS-a, której autorem także jest twórca omawianego projektu. Niemniej

jednak, realizując taki pomysł, będący prawdopodobnie periodykiem w świecie Amig NG, mógł postarać się o lepszą oprawę. Można było np.: wypalić to w formie obrazu płyty, dorzucić jakieś logo, opis zawartości, lub choćby jakąś listę i wszystko to zawrzeć w owym obrazie. Owszem taki spis jest, ale przy opisie archiwum na stronie internetowej. Jednym słowem, prócz zawartości, zabrakło tego wszystkiego, co można spotkać w klasycznych cover dyskach. A szkoda, gdyż wydając taką płytkę, można by było spróbować zarobić kilka

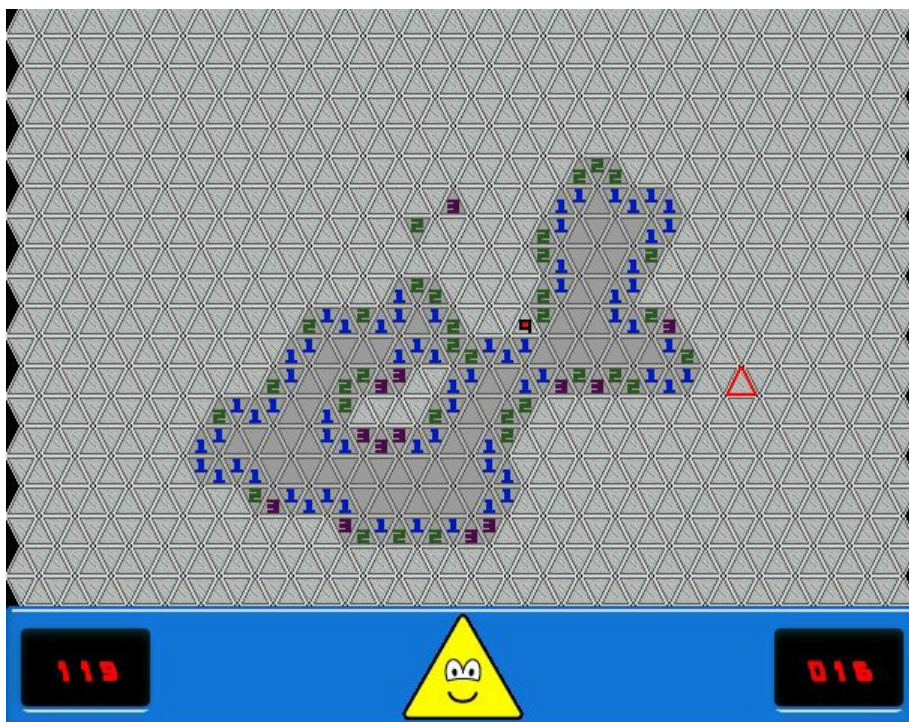
groszy. Postamigowcy, a w szczególności osoby, które wywodzą się z użytkowników linii klasycznej, kochają wszelkie wersje pudełkowe. Przypominają im one stare dobre czasy, w których to na półce nad Amigą stały skrzętnie poukładane opakowania po grach.

Choć sam pakiet, ma już parę dobrych lat i od tego czasu nikt prawdopodobnie nie szarpnął się na podobne przedsięwzięcie, to mimo wszystko liczę na to, że powstaną kolejne składanki, w których to autorzy postarają się nieco bardziej o to, aby użytkownicy, gracze, mimo doby wszechobylskiego internetu, poczuli magię starych czasów, kiedy to z wypiekami na twarzy będą wkładać kolejną płytkę do komputera, a po odpaleniu przywita ich jakieś fajnie zrobione logo ze spisem i podkładem muzycznym. Tego wszystkim Wam życzę. Póki co łapcie za joystick i miłej zabawy.

Tytuł: Aspire_games_1
Platforma: AROS
Firma: Nikos Tomatsidis
Rok produkcji: 2012
Ocena: 5



XRick



Tri Mines

FriendUP

Friend Unifying Platform

stefkos

Historia platformy Friend zaczęła się około roku 2008, kiedy kilku głównych programistów systemu AROS rozpoczęło prace nad Anubisem.

www.osnews.com/story/20516/Fo-rmer_AROS_Developers_Start_New_OS_Project_Much_Secrecy

Prace te miały przynieść Amidze nowy „rewelacyjny” system o amigowym „uczuciu”. Niestety brak czasu i wsparcia nie pozwolił, aby projekt przetrwał.

Doświadczenia przy pracy nad projektem nie poszły jednak na marne. Około roku 2012 programista Hogne Titlestad zaczął tworzyć projekt o nazwie Friend. W głównej mierze bazował na doświadczeniach pracy nad AROS-em oraz pracach NASA nad rozproszonym systemem operacyjnym:

ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/19850022397.pdf

Plan projektu zakładał tworzenie systemu „od góry”. Podejście tego typu wymagało wykorzystanie najlepszych istniejących technologii, budowanie w nich ostatnich warstw systemu i wraz z czasem przepisywanie niższych części na języki szybsze i lepiej zorientowane pod nasz system.

Część osób pewnie zada sobie pytanie, czemu tak? Nikt tak prze-

cież nie robi. Odpowiedź jest prosta – taka metoda pozwala na szybszą pracę nad platformą, pokazywanie ludziom tego co chcą zobaczyć. Co w amigowym świecie jak wiemy jest nie do pomyślenia.

Wiele osób zastanawia się „kto za tym stoi”. Zaczęło się od firmy Ideverket, dzisiejszej Friend Software Corporation. Powstała jakiś czas temu w Norwegii (Stavanger) i zajmowała się tworzeniem aplikacji webowych dla tamtejszego rynku. Tu leżą początki finansowania. Potem pokazując oprogramowanie zyskałmy kilka przychylnych firm, które chciały wykorzystać nasz „system w sieci”. Swego czasu wystartowaliśmy też na kickstartersze, niestety kwota, którą uzbierałmy, nie była zadowalająca. Aktualnie idziemy w ICO (Initial Coin Offering).

W firmie pracuje około 10 osób technicznych i około 20 biznesowych. Jeśli chodzi o napływ kolejnych to jest ich coraz więcej, chociaż nie wszystkich możemy wciągnąć do siebie. W sumie to na sieci można już zauważyć różne zmodyfikowane wersje.

Co do liczby użytkowników – o ile dobrze pamiętam to serwer demo ma ich aktualnie około 1000 (04/03/2018).

Jeżeli chodzi o rozwój systemu to myślę, że można go opisać jako stały. Kiedy nowa wersja? Teraz należy chyba zadać sobie pytanie, czy wersje dla klientów biznesowych, czy dla wszystkich?

Jak wiadomo klienci biznesowi dostają wszystko na „już”, natomiast ludzie, którzy sami budują paczki, mają lekkie opóźnienie. Oczywiście docelowe rozwiązanie jest takie, że i jedni i drudzy dostają to samo w tym samym czasie. Rozbicia na linie tak do końca nie ma, ponieważ cały system w wersji biznesowej oraz dla użytkowników posiada to samo. Różnią się one tylko dostępnym oprogramowaniem. Na chwilę obecna ogólnie dostępna wersja nie potrafi uruchamiać aplikacji windowsowych.

Jeżeli chodzi o koszty to nie ma żadnych, zarówno w jednym jak i drugim przypadku. To użytkownicy mają zarabiać i kupować od siebie programy, usługi, „czas procesora”, etc. Aktualizacje są darmowe, tylko niektóre moduły nie są dostępne jako open source. Część kodu jest oznakowana licencją MIT, część LGPL oraz AGPL.

Warto też wspomnieć, że ostatnią aktualizację źródeł na GitHubie robiliśmy w styczniu. W tym samym miesiącu opublikowano „mapę drogową” na kolejne lata – kończy się na IV kwartale 2021 roku:

friendup.cloud/wp-content/uploads/2018/01/Friend_Roadmap.png

W uproszczeniu podzielę tę mapę na trzy części:

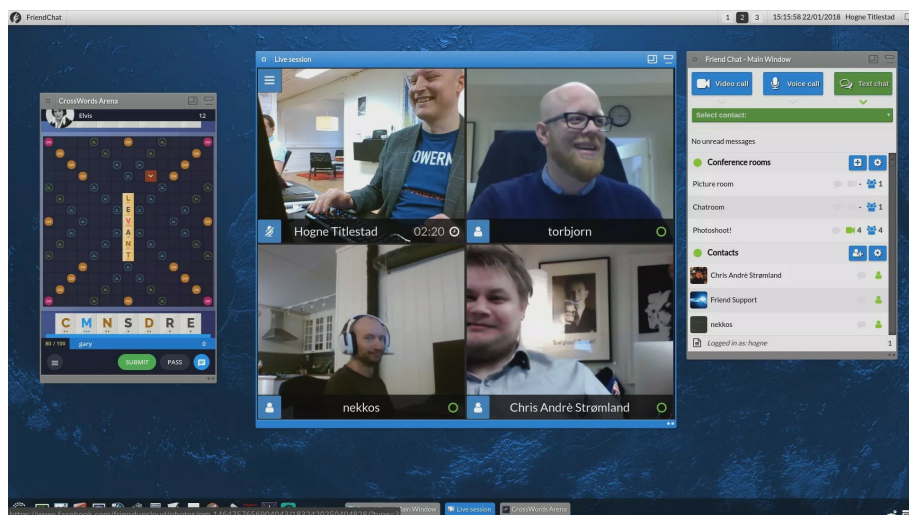
1. FriendCore (rdzeń systemu) będziemy rozszerzać o nowe systemy plików, o współdziałanie w klastrze
2. W części dotyczącej Workspace (przestrzeni roboczej) będziemy pracować nad nowymi aplikacjami, nowym GUI, siecią Friend P2P, Friend Store, wsparciem dla VR, etc.
3. Ostatnia część to „blockchain”, gdzie chcemy właściwie wciągnąć tę technologię do FriendUP-a.

O technologii „blockchain” można poczytać choćby tutaj:

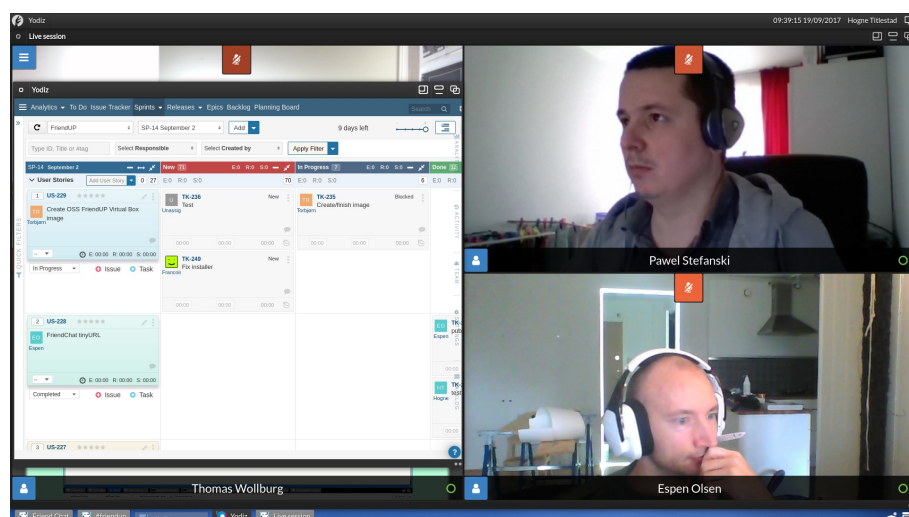
pl.wikipedia.org/wiki/Blockchain

Sprawa z oprogramowaniem wygląda bardzo dobrze i myślę, że będzie jeszcze lepiej. Aktualnie żyjemy w czasach gdy wszystko pisze się „pod sieć” i myślę, że ten trend się nie zmienia.

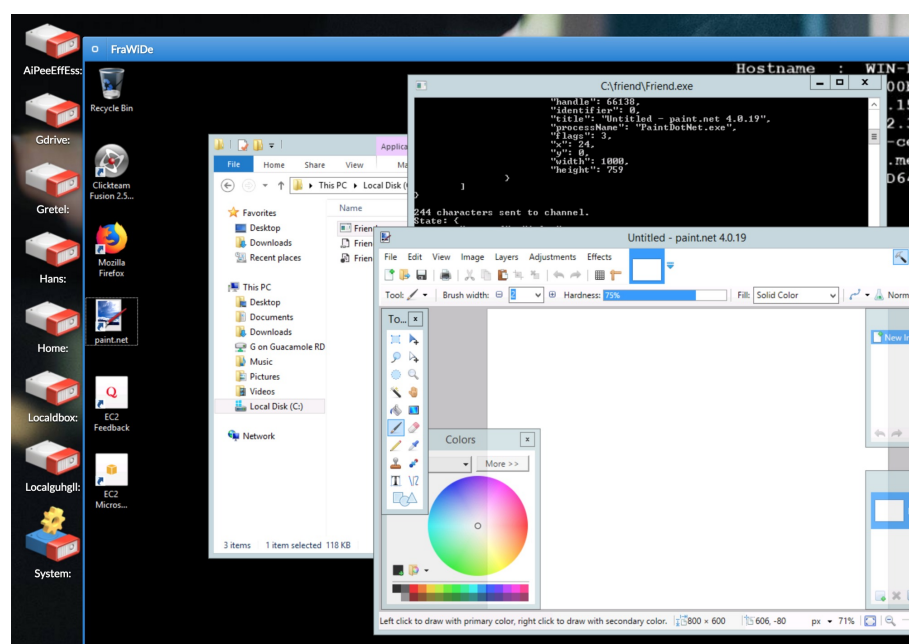
Na pewno wielu z was interesuje coś jest tak rewolucyjnego w tym systemie, że warto nim się zainteresować? Najbardziej rewolucyjne według mnie jest stworzenie płynnego oprogramowania. Działanie takiego oprogramowania wygląda tak, że użytkownik loguje się do maszyny A, robi tam coś, i nie zapisując dokumentu przesuwa to oprogramowanie na drugi pulpit uruchomiony z maszyny B. Powiem szczerze, że takiej funkcjonalności nie widziałem jeszcze w żadnym systemie operacyjnym. Są oczywiście też inne rzeczy, jak np FUI (Friend User Interface), no i chyba sam desktop. Aktualnie nie kojarzę by było gdzieś podobne



Gra crosswords i Friend Chat z konferencją 4 osób



Konferencja we Friend Chat a w nim uruchomiona aplikacja Yodiz do zarządzania projektem (tak można dzielić się aplikacjami)



Liberator – czyli możliwość odpalania windowsowych aplikacji, zdalnego desktopa

rozwiązanie na sieci, które spaja inne technologie, no ale to już chyba nie rewolucja.

Sporo mówiło się o „amigowych inspiracjach”. Co jest w FriendUP amigowego?

Workspace przypomina Workbench, jest „ściągnięcie ekranów”, FriendCore posiada biblioteki, dyski i ich obsługę, shell. Myślę że jeszcze by długo wymieniać, dlatego zapraszam do spróbowania i rejestracji!

Z amigowych ciekawostek mogę napisać iż są u nas:

Hogne Titlestad – programista AROS-a

François Lionet – twórca AMOS-a

David Pleasance – główny manager Commodore UK

Paul Lassa – współtwórca Amigi 4000 oraz jeżeli dobrze pamiętam Commodore 65

Dan Wood – znany amigowy bloger

Jak zacząć swoją przygodę z FriendUP? Są dwie możliwości.

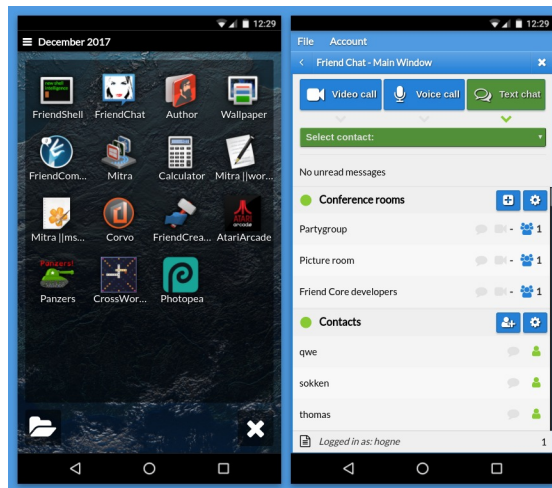
Można ściągnąć źródła z githubowego repozytorium i postawić sobie serwer u siebie w domu:

github.com/FriendUPCloud/friendup

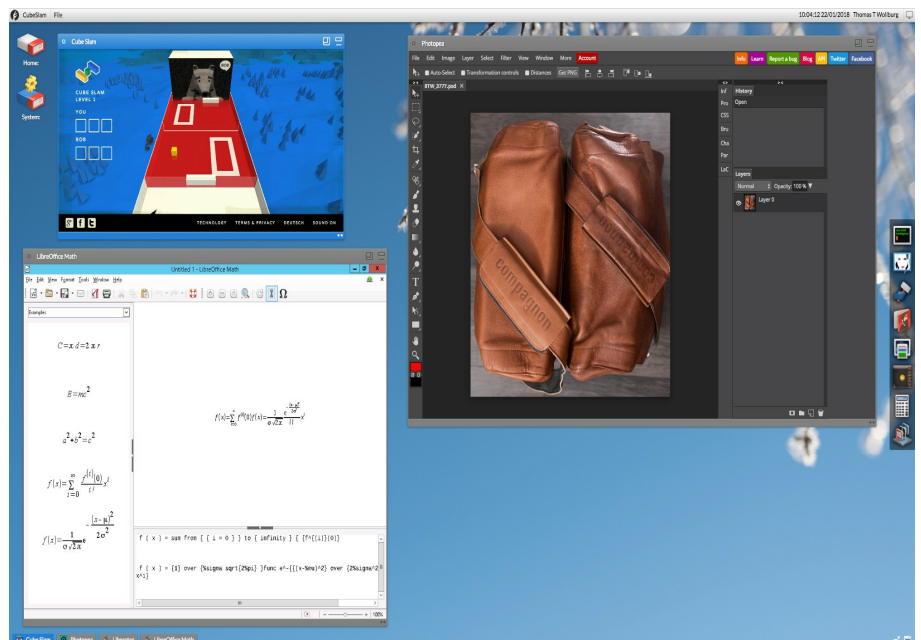
Można też wypełnić formularz prosząc o dostęp do wersji demo:

friendup.cloud/demo/

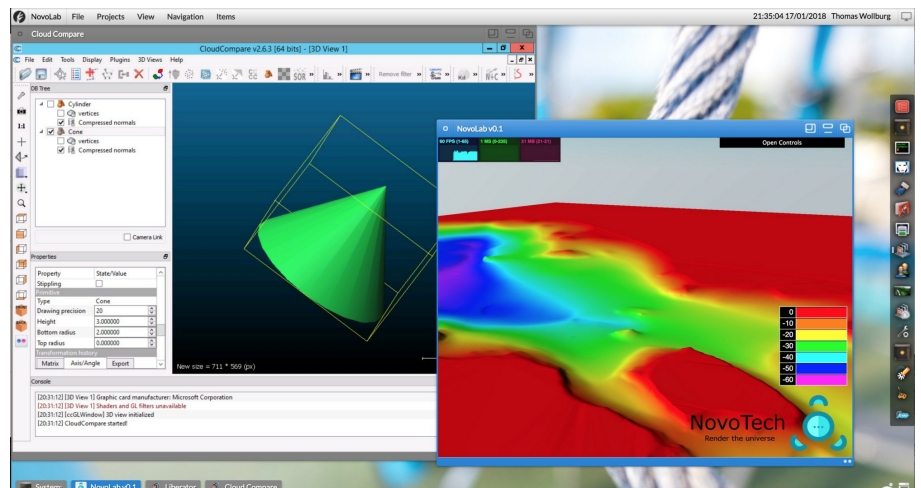
Gorąco zachęcam do wypróbowania „Friendu”. System działa z poziomu przeglądarki, więc jest niekłopotliwy, procedura rejestracji jest szybka – myślę, że warto spróbować.



Android – po lewej Workspace, po prawej Friend Chat



Club Slam (jak widać taki „Pong”), po prawej edytor Photopea (napisana w JavaScript) i do tego LibreOffice; uruchomione przez Liberatora



Liberator a w nim aplikacja CloudCompare

Dział „Nowości” w tym numerze zawiera wyjątkowo mało obrazków. Jest to pochodna przejścia na druk – liczba stron musi być podzielna przez 4. Jednak – mając na uwadze dobro czytelników – trochę ilustracji da się wcisnąć (kosztem ilustracji zapowiedzi).

Kolejno są to: *OpenJazz*, *TCGS Cave*, *Stunt Car Racer Remake*, *Wyrmsun*.



Co w przyszłym numerze

AmigaOS 4

- szkółka DrawStudio cz. 2 (i ostatnia, w planach kolejne programy graficzne)
- szkółka Hollywood cz. 4 (ubajerzanie zegara)

MorphOS

- wybór komputera Apple
- flaszowanie kart graficznych

AROS

- AmiBridge – integracja „klasyka” z AROS-em

ERRATA

Pojawiła się nowa wersja pluginu APK Compiler, o którym był artykuł w poprzednim numerze magazynu. W wersji 2.1 nie trzeba już instalować JDK ani Gradle, wystarczy sam Android Studio. Właściwie to nie tyle sprostowanie błędu, co dobra wiadomość.

Niestety nie da się już skompilować APK-i dla Androida w wersji starszej niż 4.0.

REKLAMA



LIBGL: FBO are in core, and so used
 LIBGL: PointSprite are in core, and so used
 LIBGL: CubeMap are in core, and so used
 LIBGL: BlendColor is in core, and so used
 LIBGL: Blend Substract is in core, and so used
 LIBGL: Blend Function and Equation Separation is in core, and so used
 LIBGL: Extension GL_OES_element_index_uint detected
 LIBGL: high precision float in fragment shader available and used
 LIBGL: Extension GL_EXT_frag_depth detected and used
 LIBGL: Max vertex attrib: 16
 LIBGL: Max texture size: 16384

Rotate Image PNG lib EGL for OpenGL ES AmigaOS4



Window mode activated for EGLwrap lib (default) | Lister video mode: disabled (default) | EGLwrap bpp: 32 bits

General | 1 Settings | 2 Internal texture settings | 3 Lowlevel | 3 About

OpenGL ES 2.0 **EGLWrap**

EGLWRAP GUI 0.0.9 (24.04.2017)
 Using GUI-EGLWrap created for AmigaOS4 by Hugues "HunoPPC" Nouvel
 HunoPPC Amiga French Team (Bourgoigne), Amiga Makes it possible...
 EGLWrap library is a wrapper EGL for OpenGL ES library NOVA AmigaOS4 by A-EON 2017

Load default prefs | Quit GUI-EGLWrap

